UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

MODELACIÓN DE LA TRANSFERENCIA DE CALOR EN UN INVERNADERO USANDO FVM

TESIS

QUE PARA OPTAR POR EL GRADO DE

MAESTRO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

PRESENTA:

JOSÉ ANTONIO BORRAS GUTIÉRREZ

Director de Tesis:

DR. LUIS MIGUEL DE LA CRUZ SALAS

Instituto de Geofísica, UNAM

Ciudad Universitaria, CDMX , Enero 2020

Agradecimientos

Quisiera comenzar agradeciendo a mi asesor de tesis el Dr. Luis Miguel De la Cruz Salas por todo su tiempo invertido en este trabajo y toda la orientación que me brindó para el mismo. Fue una fortuna contar con un tutor que genuinamente se preocupa por sus alumnos y sus intereses.

En el mismo sentido quiero agradecer a todos mis sinodales: Dr. Christopher Rhodes Stephens Stevens, Dra. María Elena Lárraga Ramírez, Dr. Oscar Alejandro Esquivel Flores y Dr. Juan Manuel Hernández Calderon. A ellos les debo que hayan contribuido con su visión, sus críticas y sugerencias que fueron de mucha ayuda para mejorar mi trabajo.

Al coordinador del posgrado, el Dr. Javier Gómez Castellanos, le agradezco muchísimo todo el apoyo y la paciencia brindados.

También, debo agradecer a toda la genete que conocí en el posgrado: incluyendo docentes, compañeros y administrativos por toda la solidaridad, respeto y buen humor.

Por supuesto, debo de agradecer al CONACyT por el apoyo económico brindado y sin el cual no habría podido realizar una maestría.

Agradezco especialemte a mi madre Rosario y a mi novia Ferrara por todo el apoyo emocional que me brindaron en cada crisis de fin de semestre y en general a todas esas personas con las que puedo contar: mi padre, mis hermanos, mis amigos y amigas que a pesar de mis errores jamás me han abandonado.

Gracias de verdad!

Índice general

Capítulo 1

Introducción

En el año 2015, la Organización de las Naciones Unidas (ONU) lanzó un conjunto de objetivos globales para erradicar la pobreza, proteger el planeta y asegurar la prosperidad para todos como parte de una nueva agenda de desarrollo sostenible[1]. El segundo de estos objetivos es: "Poner fin al hambre, lograr la seguridad alimentaria y la mejora de la nutrición y promover la agricultura sostenible"[2]. Una de las formas de atacar este problema es mediante el uso de invernaderos pues se trata de un ambientes controlados que pueden suministrar condiciones ideales para cada tipo de cultivo y obtener así una mayor producción por metro cuadrado y durante todo el año . Así, en un invernadero el manejo de las variables climáticas (la temperatura, la humedad, la iluminación, etcétera) es una de las más importantes tareas para maximizar la calidad y cantidad de los cultivos [4]. Por otro lado, la dinámica de fluidos computacional (CFD por sus siglas en inglés) permite realizar simulacros computacionales del comportamiento de los fluidos cuando se someten a ciertas condiciones físicas. Esta área de estudio ha ido mejorando su capacidad de predicción, en parte, gracias al aumento del poder computacional que sea ha dado en las últimas décadas. El presente trabajo tiene como principal motivación utilizar CFD para analizar la distribución de dichas variables climáticas en un invernadero, en particular la tempuratura y ventilación y ,de hecho, en años recientes se han realizado trabajos académicos que se valen de CFD para estudiar la climatización de invernaderos[4]-[9] (ver la sección 1.2 para más información acerca de los antecedentes). Sin embargo, se trata de un tema que aún sigue planteando nuevas interrogantes pues son muchos los factores que entran en juego en este tipo de investigaciones y cada invernadero tiene sus propias particularidades. Además la complejidad del problema no se limita al planteamiento físico sino que también presenta un importante reto computacional. Esta alta demanda computacional no es exclusiva de los invernaderos y más bien es típica en la computación científica o modelación, por lo que con el desarrollo de una implementación propia, reutilizable pero orientada al alto desempeño, se busca contribuir al conocimiento en estas áreas.

Como hemos dicho, el presente trabajo de tesis pretende aportar conocimientos relevantes para la planeación y mantenimiento de invernaderos, lo que impacta directamente en la viabilidad de los proyectos de invernaderos, además de contribuir con la democratización de software, es decir, con proveer código que será de fácil acceso. La siguiente sección precisa los alcances del trabajo.

1.1. Objetivos y Metas

Con la intención de trabajar geometrías en su mayoría cartesianas se optó por modelar la transferencia de calor en un modelo tipo edificio. En particular, el presente trabajo se inclinó por los invernaderos debido a dos motivos; por un lado, existen ya textos académicos en este sentido que prueban el interés en este tipo de simulaciones y, por el otro, el software desarrollado tendría el potencial para aplicaciones de diseño y/o monitoreo. De esta manera, el objetivo general es el de **estudiar distintos escenarios de transferncia de calor en un invernadero mediante simulaciones generadas por medio de una implementación computacional basada en el método de volumen finito (FVM por sus siglas en inglés) que sea propia y accesible**. Este objetivo general origina metas más puntuales, a continuación se enlistan las metas a realizar en el presente trabajo de tesis.

- *Producir un software accesible.* En la medida de lo posible, el código debe estar orientado a facilitar la colaboración, modificación y utilización del mismo.
- *Obtener una implementación versátil de FVM.* Aunque inspirado en un problema en particular, el software debe manejar problemas con distintas situaciones físicas (1 dimensión, convección forzada, etcétera). Esto también permite que el software sea reutilizabe.
- *Obtener simulaciones de invernaderos en distintas geometrías.* Parte de la planeación de los invernaderos consiste en la geometría de los mismos por lo que es importante obtener resultados en distintas geometrías, en particular la forma del techo o la posición de ventilaciones.

1.2. Antecedentes

En la redacción de cualquier texto académico es importante poder contextualizar el estudio a realizar, es decir, tener claro qué trabajos se han hecho anteriormente que aborden temas similares o relacionados. A continuación presentamos, sin ningún orden específico, algunos de los trabajos revisados mencionando brevemente los puntos más relevantes asociados a este estudio.

1.2.1. Modelación térmica de invernaderos

Analizar como impactan las distintas geometrías de un invernadero en la temperatura del aire dentro del mismo es una idea que se analiza en [10]. En este artículo, se modela al invenadero con una simulación en dos dimensiones (2D); la justificación que se da a esta consideración es que los invernaderos considerados tienen entre 50m-150m de largo y un ancho de entre 10m-14m, de esta manera se puede asumir que los invernaderos solo tienen variación en 2 dimensiones. Sin embargo, a pesar de ser en 2D, no se trata de un modelo sencillo pues se considera un problema no-estacionario (que

depende del tiempo) en el que además se incluyen una gran cantidad de factores físicos entre los cuales se encuentran: radiación solar, profundidad del suelo, humedad del aire, condensación sobre las paredes, transpiración del cultivo, etc. Se toman en cuenta, además, distintos escenarios en los que se varia ligeramente la geometría del invernadero, como por ejemplo al aumentar o disminuir un par de metros al ancho del invernadero. Por otro lado, la discretización del espacio se hace usando $60,000$ celdas. Esto significa que el dominio continuo se divide en $60,000$ secciones distintas. La realización del análisis recién descrito lleva a los autores a concluir cuál es el mejor diseño, en términos geométricos, para la preservación de temperatura dentro del invernadero aunque, por supuesto, un mayor respaldo experimental que valide la conclusión es requerido.

Otro estudio en 2D que examina el compartamiento térmico de un invernadero puede revisarse en [6]. En dicho estudio la simulación se realiza utilizando el método de volumen finito con una malla estructurada de $20,000$ celdas. La geometría del techo es tipo arco y las dimensiones son 8m de ancho por 20m de largo y una altura máxima de 4.1m, el modelo solo considera el ancho y alto del invernadero (2D). Además, el modelo conceptual considera la radiación que incide en el invernadero a lo largo del día por lo que parte importante de la discusión se centra en la creación de una función que representara la radiación incidente en el invernadero durante un lapso de 8h, de acuerdo a la geolocalización del invernadero. Las simulaciones se realizan para dos casos; uno suponiendo una temperatura constante al exterior del invernadero y otro suponiendo una temperatura variable. Dentro de sus conclusiones está la afirmación de que el mecanismo básico para la transferencia de calor, en este caso, es la convección resultante del aire que entra al invernadero excepto en las esquinas donde la radiación influye en la temperatura.

Más recientemente, en [4] se realiza un estudio que ya ocupa un modelo de tres dimensiones (3D) en el que se analizan los efectos que tienen las *pantallas de sombreado* en el clima del invernadero. Estas *pantallas de sombreado* son un tejido hecho con fibras de plástico con el objetivo de proteger el cultivo de la radiación solar pero que inciden también en la ventilación del invernadero; para observar este último efecto, se considera dentro del modelo que las pantallas se comportan como un medio poroso con caracterísitcas estudiadas previamente en un laboratorio. Como método númerico para dar solución a las ecuaciones del modelo se usa el método de volúmen finito ocupando distintas mallas definidas con un número de volúmenes N que van desde los 2×10^5 hasta los 3×10^6. Adicionalmente la simulación es estacionaria, de un invernadero vacío (sin tomar en cuenta los efectos del cultivo) y usando un flujo turbulento. De la comparación entre el escenario con *pantallas* y sin *pantallas* se observa claramente que la presencia de *pantallas* afecta significativamente no sólo los patrones de flujo sino también la magnitud de la velocidad. De lo anterior, los autores concluyen que aunque los resultados dependen la geometría del invernadero y la posición de las *pantallas* los resultados remarcan que el efecto de dichas *pantallas* en las condiciones de ventilación no debe ser ignorado para proveer resultados realistas.

Otro antecedente que vale mencionar es el estudio realizado en [7], pues las

condiciones de frontera de este trabajo de tesis están basadas en dicho estudio. En el artículo se modelan 3 distintos invernaderos con geometrías en 2D usando dos métodos numéricos distinos: volumen finito (FVM) y elemento finito (FEM). De acuerdo a lo mencionado en [7], FVM es versátil e intuitivo, relativamente fácil de programar y como resultado es el método más comunmente usado, particularmente para estudiar la ventilación en invernaderos. Sin embargo, en situaciones con geometría o condiciones de frontera complicadas FEM es frecuentemente usado aunque existen pocos paquetes comerciales disponibles, debido a las dificultades en la implementación de esta técnica. Uno de los objetivos principales de la investigación citada es contrastar las diferencias existentes entre ambas técnicas para este caso de uso (la modelación de la ventilación en invernaderos) y en este sentido se concluye que en la mayoría de los casos los métodos concuerdan mejor en las tempertaturas que en las velocidades aunque cualitativamente los flujos fueron similares. Es muy importante hacer notar que este artículo discute también las diferencias computacionales entre FVM y FEM; al respecto se encuentra que, en promedio, FEM requiere el doble de tiempo de cómputo por celda y aproximadamente 10 veces más almacenamiento en memoria con respecto a FVM.

Desde otra perspectiva, la simulación numérica de la distribución de temperatura en invernaderos ventilados naturalmente, está relacionada a un tipo de problema de mecánica de fluidos conocido como *convección natural*. De esta manera, resulta valioso examinar también los antecedentes que existen en la investigación de la convección natural.

1.2.2. Convección natural

La convección natural es un fenómeno de la mecánica de fluidos que se caracteriza porque el campo de velocidad del fluido no es supuesto sino que se calcula considerando el efecto de una temperatura que no es constante en el espacio. Por ejemplo, en modelos donde el movimiento del fluido es generado por una fuente externa como ventiladores o bombas la velocidad se puede incluir como un supuesto pero en la *convección natural* esta debe obtenerse de las ecuaciones como consecuencia de la temperatura. A continuación se discuten algunos de los antecedentes en la solución numérica de la *convección natural* para distintos escenarios. Existen una gran cantidad de artículos que tratan el problema de la convección natural en una cavidad, en este trabajo resultan relevantes 4 textos que se tomaron como base para la validación. Los primeros 3 consisten baśicamente en usar una aproximación numérica para resolver las ecuaciones adimensionales que describen la convección natural en dos dimensiones. Estos se tratan de *Natural convection of air in a square cavity: A bench mark numerical solution* [11], *Effects of thermal boundary conditions on natural convection flows within a square cavity* [12] y *Numerical investigation of natural convection in a rectangular enclosure due to partial heating and cooling at vertical walls* [13]. En el primero se plantean las condiciones de frontera más simples, en el segundo se plantea una condición de

frontera que es función de la posición y en el tercero se usan paredes que son calentadas o enfriadas parcialmente. Por último en *Natural convection in a cubic cavity: implicit numérical solution of two benchmark problems*[14] se resuelven condiciones de frontera sencillas pero para una cavidad en tres dimensiones.

1.3. Hipótesis

El planteamiento desarrollado hasta el momento lleva implícito ciertas afirmaciones o hipótesis que pudieran comprobarse o refutarse con los resultados del trabajo. Estas hipótesis son:

- La distribución de temperatura resultante de las ecuaciones es susceptible a la geometría del techo del invernadero.
- La posición de las ventilaciones en un invernadero impacta de forma cuantitativa el flujo de calor.
- La aproximación de Boussinesq y en general la metodología propuesta es suficiente para resolver los modelos planteados.
- El lenguaje de programación *Python* tiene las herramientas necesarias para realizar cómputo de alto rendimiento.

1.4. Organización de la Tesis

El texto está dividido en 5 secciones principales: *Introducción*, *Marco Teórico*, *Desarrollo de Software y primeras pruebas*, *Modelación de la transferencia de calor en un invernadero* y *Conclusiones*. En esta primera sección, *Introducción*, se exponen las motivaciones y objetivos, así como otros estudios que se han desarrollado en temas similares. En la sección, *Marco Teórico*, se puede revisar la teoría en la que está basado el estudio; más específicamente, el modelo conceptual (sección 2.1), matemático (sección 2.2) y numérico(sección 2.3). La tercera sección se divide en dos partes, en la primera se discuten los puntos claves de la forma en que está constituido el software (modelo computacional) y en la segunda se presentan los primeros ensayos en problemas de convección natural. Estas dos partes corresponden a las secciones 3.2 y 3.3, esta última constituye un puente entre el desarrollo y los resultados pues las soluciones que se muestran representan resultados preliminares o que sirven para validar los métodos desarrollados para la convección natural. Los resultados más relevantes se exponen en la sección *Modelación de la transferencia de calor en un invernadero*, aquí se muestran los resultados de simulaciones considerando 3 tipos de geometría de techo combinadas con 2 tipos de ventilaciones. En *Conclusiones*, como es habitual, se recapitula la información repasando algunos de los puntos clave del trabajo.

Capítulo 2

Marco Teórico

El presente trabajo está basado en un modelo específico del comportamiento físico de los fluidos. En este capítulo se discute brevemente dicho modelo, así como otros conceptos y suposiciones relevantes, con la finalidad de establecer el contexto teórico de esta tesis. En particular, se presentan el modelo conceptual, modelo matemático y modelo numérico que definen el marco de trabajo.

2.1. Modelo Conceptual: Abstracción del invernadero

Para representar la transferencia de calor en un invernadero, se debe realizar una conceptualización de la situación física real que se traduce en un planteamiento formal, formal desde el punto de vista matemático. La forma de realizar esta abstraccón o modelo conceptual no es un proceso obvio y constituye un reto en sí mismo. Como una primera aproximación al problema se consideró vacíos (a excepción del aire) a los invernaderos simulados en este estudio. Esta primera estimación tiene además el objetivo de obtener conclusiones independientes del tipo de cultivo y de evidenciar más claramente el efecto que tiene la geometría de los invernaderos en los resultados obtenidos. Para los alcances del presente trabajo, de ahora en adelante se va a representar al invernadero con una figura de dos dimensiones inmersa en un dominio rectangular. En el trabajo *'Comparison of finite element and finite volume methods for simulation of natural ventilation in greenhouses'*[7] se utilizan representaciones de este estilo y en paticular, el que se muestra en la figura 2.1. Es por esto que se tomó el texto recién citado como base para la creación del modelo conceptual utilizado en el presente trabajo. Dicho modelo conceptual se define al asumir que el invernadero tiene una temperatura fija Tr en su parte superior (techo), paredes laterales adiabáticas ($f_{wall} = 0$) y una temperatura T_s (mayor que Tr) en el suelo del invernadero. El suelo que se encuentra fuera del invernadero y las demás paredes del dominio se modelan con una temperatura fija T_a. Cómo hemos dicho, estas suposiciones están extraídas del modelo conceptual de [7]. A pesar de que en dicho modelo no se incluye la radiación, se decidió incluir las mismas consideraciones pues, según los mismos autores, con estas idealizaciones se podría imitar la absorción de la radiación solar en la superficie del suelo del invernadero. Para los objetivos del presente trabajo es importante además estudiar el papel de la

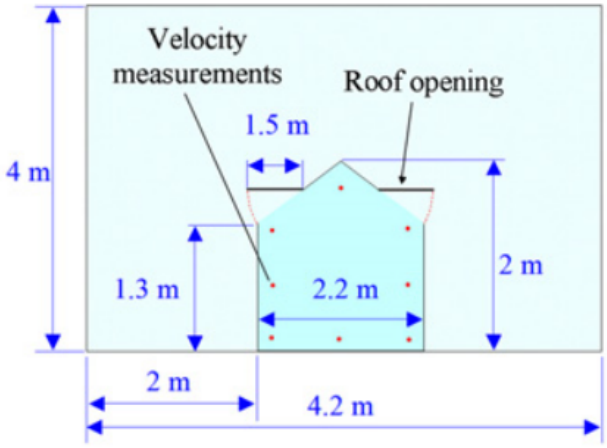


FIGURA 2.1: Esquema del dominio y condiciones de frontera usado en [7], donde $Ta = 18.4^oC, Tr = 21.2^oC, f_{wall} = 0W/m^2, f_{soil} = 74.7W/m^2$ y $P_0 = P_{atm}$.

ventilación en la distribución de temperatura por lo que, las figuras usadas para representar el invernadero cuentan con una abertura que se ubica, ya sea sobre la pared derecha del invernadero o sobre el techo del mismo. Se considera además que la ventilación y el resto de fronteras tienen una presión fija P_o igual a la presion atmosférica. Distintas figuras se probaron para representar los invernaderos y realizar las simulaciones. A la definición formal de dichas figuras junto sus condiciones de frontera se les denominará: *configuración*. Las configuraciones varían en la forma geométrica de la sección que representa el techo del invernadero y de la sección que representa la ventilación (abertura) aunque todos comparten los parámetros de la tabla 2.1 que se usan para definir las condiciones de frontera.

En la mecánica de fluidos es común el uso de ecuaciones y variables adimen-

Parámetro	magnitud real	magnitud adim
Tr	20.2^oC	0.439
Ta	18.4^oC	0.0
Ts	22.5^oC	1.0
f_{wall}	$0 \quad W/m^2$	0.0
P_o	1 atm	1.0

TABLA 2.1: Parámetros y su respectivo valor que definen las condiciones de frontera de todos los modelos utilizados para las simulaciones obtenidas.

sionales por lo que en la tabla 2.1 se presentan también el valor del parámetro en su forma adimensional (*magnitud adim*). Además de las suposiciones antes mencionadas, se considera que la transferencia de calor se da por medio de la conducción y de la convección y que la distribución de temperatura se

encuentra un estado estacionario. El incluir tanto la conducción como la convección en el modelo, así como estudiar la situación estacionaria tiene una conexión directa al modelo matemático. A continuación se describe el modelo matemático y las repercusiones que premisas como estas tienen en las ecuaciones.

2.2. Modelo Matemático: Convección natural en su forma adimensional

2.2.1. Ecuaciones de Navier-Stokes

Para la simulación de la transferencia de calor en un invernadero, se tomarán como base las ecuaciones de Navier-Stokes para un fluido newtoniano. De acuerdo con [16], y usando t para representar el tiempo, estas ecuaciones en coordenadas cartesianas (x, y, z), tienen la siguiente forma:

$$\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \vec{\mathbf{u}}) = 0 \tag{2.1}$$

$$\frac{\partial (\rho u)}{\partial t} + \boldsymbol{\nabla} \cdot (\rho u \vec{\mathbf{u}}) = -\frac{\partial p}{\partial x} + \boldsymbol{\nabla} \cdot (\mu \boldsymbol{\nabla} u) + S_{Mx} \tag{2.2}$$

$$\frac{\partial (\rho v)}{\partial t} + \boldsymbol{\nabla} \cdot (\rho v \vec{\mathbf{u}}) = -\frac{\partial p}{\partial y} + \boldsymbol{\nabla} \cdot (\mu \boldsymbol{\nabla} v) + S_{My} \tag{2.3}$$

$$\frac{\partial (\rho w)}{\partial t} + \boldsymbol{\nabla} \cdot (\rho w \vec{\mathbf{u}}) = -\frac{\partial p}{\partial z} + \boldsymbol{\nabla} \cdot (\mu \boldsymbol{\nabla} w) + S_{Mz} \tag{2.4}$$

$$\frac{\partial (\rho i)}{\partial t} + \boldsymbol{\nabla} \cdot (\rho i \vec{\mathbf{u}}) = -p \boldsymbol{\nabla} \cdot \vec{\mathbf{u}} + \boldsymbol{\nabla} \cdot (\kappa \boldsymbol{\nabla} T) + S_i \tag{2.5}$$

Donde i reprerenta la energía interna, $\vec{\mathbf{u}} = u\hat{\imath} + v\hat{\jmath} + w\hat{k}$ representa la velocidad de flujo; mientras que ρ, p, μ, κ son la densidad, la presión, viscosidad y difusividad térmica, respectivamente. En nuestro modelo consideraremos que $i = C_v T$ con C_v el calor específico a volumen constante y T la temperatura. Por último, S_M y S_i denotan las fuentes de momento y de energía interna, respectivamente. En el caso más general, $\vec{\mathbf{u}}$ es un campo vectorial y el resto de variables, son campos escalares. Más precisamente, en el caso general se tiene que $u = f(x, y, z, t)$, $v = g(x, y, z, t)$ y $w = h(x, y, z, t)$; dónde $f : \mathbb{R}^3 \times \mathbb{R} \longrightarrow \mathbb{R}$, $g : \mathbb{R}^3 \times \mathbb{R} \longrightarrow \mathbb{R}$ y $h : \mathbb{R}^3 \times \mathbb{R} \longrightarrow \mathbb{R}$ son las respectivas funciones. Así, $\vec{\mathbf{u}} = u\hat{\imath} + v\hat{\jmath} + w\hat{k} = F(x, y, z, t)$ con $F : \mathbb{R}^3 \times \mathbb{R} \longrightarrow \mathbb{R}^3$. El resto de variables tienen el mismo dominio y codominio que f.

2.2.2. Convección natural con aproximación de Boussinesq

El modelo recién descrito es un modelo muy general, sin embargo, el problema que se plantea lleva consigo una serie de suposiciones al modelo matemático que, entre otras cosas, eliminan algunas dependencias. Estas premisas y sus relaciones con el modelo matemático se describen a continuación.

- Dado que el aire es el único fluido en el dominio tenemos que el flujo es monofásico.
- El flujo es estacionario, es decir, ya no se supone que u es función de las coordenadas y del tiempo sino simplemente es función de las coordenadas. De hecho, la suposición es que se ha alcanzado un equilibrio y **ninguna de las funciones** depende del tiempo.
- El fluido es incompresible, esto significa que la función que define a ρ es ahora una función constante, por lo que la ecuación(2.1) se simplifica a $\boldsymbol{\nabla} \cdot (\vec{\mathbf{u}}) = 0$.
- Al igual que ρ, se supone que: μ, C_v y κ no dependen tampoco de la posición es decir que son funciones constantes.
- El fluido no tiene turbulencia.
- El movimiento ocurre en el plano XY, es decir, se supone que ninguna función depende de z y que $z = w = 0$.
- La única fuente de momento (externa al cuerpo) se da en el sentido Y debido a la gravedad. Matemáticamente $S_{Mx} = S_{Mz} = 0$ y $S_{My} = -\rho g$. Donde $g \in \mathbb{R}$ es una función constante que representa la magnitud de la aceleración de la gravedad en el dominio.
- Se considera que en el término ρg la densidad no debe ser considera constante sino que puede aproximarse usando una diferencia de temperaturas.

Este último punto se conoce como la aproximación de Boussinesq y consiste en hacer la sustitución $\rho g = \rho_0 g[1 - \beta(T - T_0)]$ en el último término de la ecuación de momento en 'Y' (ecuación 2.3). Donde $\beta \in \mathbb{R}$ representa el coeficiente de expansión volumétrica, $\rho_0 \in \mathbb{R}$ y $T_0 \in \mathbb{R}$ son valores de densidad y temperatura de referencia, respectivamente, que además se consideran constantes. Dado que T_0 y ρ_0 son valores de referencia pueden tomar cualquier valor constante aunque su valor específico pierde relevancia cuando se obtienen ecuaciones adimensionales. De esta manera la ecuaciones a resolver son finalmente:

$$\boldsymbol{\nabla} \cdot (\vec{\mathbf{u}}) = 0 \tag{2.6}$$

$$\boldsymbol{\nabla} \cdot (\rho_0 u \vec{\mathbf{u}}) = -\frac{\partial p}{\partial x} + \boldsymbol{\nabla} \cdot (\mu \boldsymbol{\nabla} u) \tag{2.7}$$

$$\boldsymbol{\nabla} \cdot (\rho_0 v \vec{\mathbf{u}}) = -\frac{\partial p}{\partial y} + \boldsymbol{\nabla} \cdot (\mu \boldsymbol{\nabla} v) - \rho_0 g[1 - \beta(T - T_0)] \tag{2.8}$$

$$\boldsymbol{\nabla} \cdot (\rho_0 C_v T \vec{\mathbf{u}}) = \boldsymbol{\nabla} \cdot (\kappa \boldsymbol{\nabla} T) \tag{2.9}$$

Las ecuaciones recién descritas representan de forma completa el comportamiento de un fluido en las condiciones dadas. En atención a lo expuesto, las únicas variables que no son constantes sino que dependen sólo de la posición son: u, v, p y T. Además de la velocidad $\vec{\mathbf{u}}$ pues está definida en términos de

sus componentes cartesiandas u y v. Este modelo se conoce como *convección natural con la aproximación de Boussinesq* y en los estudios que lo abordan es común la definición de variables adimensionales para simplificar el planteamiento matemático de las ecuaciones. De acuerdo con [13], se deben definir las siguientes variables adimensionales:

$$X = \frac{x}{L}, \quad Y = \frac{y}{L}, \quad U = \frac{Lu}{\kappa}, \quad V = \frac{Lv}{\kappa}, \quad P = \frac{pL^2}{\rho\kappa^2}, \quad \theta = \frac{T - T_0}{T_1 - T_0},$$

Así como los números de Prandtl (Pr) y de Rayleigh (Ra):

$$Pr = \frac{\mu}{\kappa}, \quad Ra = \frac{g\beta(T_1 - T_0)L^3}{\mu\kappa}$$

Donde L, T_1 y T_0 son constantes. L representa una longitud característica mientras que T_1 y T_0 son temperaturas de refrencia (como las temperaturas alta y baja en paredes opuestas). De este modo, Pr y Ra quedan como constantes, X, Y quedan como variables independientes y U, V, P, θ son las nuevas variables dependientes. Es decir, $U = U(X,Y)$, $V = V(X,Y)$, $P = P(X,Y)$, $\theta = \theta(X,Y)$. Entonces, las ecuaciones que describen la *convección natural* quedan expresadas en términos de las variables adimensionales de la siguiente manera:

$$\frac{\partial U}{\partial X} + \frac{\partial V}{\partial Y} = 0 \tag{2.10}$$

$$U\frac{\partial U}{\partial X} + V\frac{\partial U}{\partial Y} = -\frac{\partial P}{\partial X} + Pr\left[\frac{\partial^2 U}{\partial X^2} + \frac{\partial^2 U}{\partial Y^2}\right] \tag{2.11}$$

$$U\frac{\partial V}{\partial X} + V\frac{\partial V}{\partial Y} = -\frac{\partial P}{\partial Y} + Pr\left[\frac{\partial^2 V}{\partial X^2} + \frac{\partial^2 V}{\partial Y^2}\right] + PrRa\theta \tag{2.12}$$

$$U\frac{\partial \theta}{\partial X} + V\frac{\partial \theta}{\partial Y} = \left[\frac{\partial^2 \theta}{\partial X^2} + \frac{\partial^2 \theta}{\partial Y^2}\right] \tag{2.13}$$

O equivalentemente, si sumamos la ecuación(2.10)+ecuación(2.11), la ecuación(2.10)+ecuación(2.12) y la ecuación(2.10)+ecuación(2.13) al reacomodar y definir $X_1 = X$, $X_2 = Y$, $U_1 = U(X,Y)$, $U_2 = V(X,Y)$, $S_1 = 0$, $S_2 = PrRa\theta(X,Y)$; se obtiene:

$$\frac{\partial}{\partial X_j}(\rho U_j) = 0 \tag{2.14}$$

$$\frac{\partial}{\partial X_j}(U_j U_i) = -\frac{\partial P}{\partial X_i} + \frac{\partial}{\partial X_j}\left(Pr\frac{\partial U_i}{\partial X_j}\right) + S_i \tag{2.15}$$

$$\frac{\partial}{\partial X_j}(U_j \theta) = \frac{\partial}{\partial X_j}\left(\frac{\partial \theta}{\partial X_j}\right) \tag{2.16}$$

Es importante resaltar que las ecuaciones resultantes están expresadas en notación indicial (queda implítico que existe una suma sobre los índices repetidos). Dentro de este marco, las 2 ecuaciones de momento (3 ecuaciones para 3D) quedan expresadas por la ecuación(2.15). Una de las ventajas de expresar el modelo matemático en la forma anterior es que las ecuaciones pueden considerarse como casos particulares de una expresión más general conocida como la ecuación general de transporte. La forma de esta ecuación es la siguiente:

$$\frac{\partial}{\partial X_j}(U_j(X,Y)\phi(X,Y)) = \frac{\partial}{\partial X_j}\left(\Gamma\frac{\partial\phi(X,Y)}{\partial X_j}\right) + S(X,Y) \tag{2.17}$$

Para obtener las ecuaciones de Navier-Stokes de la ecuación de transporte se deben sustituir las variables como se muestra en la Tabla 2.2. En atención a

TABLA 2.2: Correspondencia de las ecuaciones de balance en flujo laminar con la ecuación general de transporte.

Ecuación	ϕ	Γ	**S**
Masa	ρ	0	0
Momento	U_i	Pr	$-\frac{\partial P}{\partial X_i} + b_i$
Energía	θ	1	0

lo expuesto, la ecuación general de transporte es la euación que describe el comportamiento físico de las propiedades ϕ (como temperatura) que deben encontrarse o resolverse; es decir, la ecuación2.17 es la ecuación diferencial parcial que buscamos resolver y por lo tanto, constituye la base del modelo numérico. Antes de pasar al modelo numérico, para delimitar bien el problema es necesario definir el dominio físico y las condiciones de frontera.

2.2.3. Condiciones de frontera

De forma matemática, el dominio del problema planteado es el conjunto de puntos $\Omega = [0,1] \times [0,1]$. Cómo sabemos, la diferencia de las distintas *configuraciones* es la geometría en la que están definifas las condiciones de frontera. Con el objetivo de dar una definición formal de la primer configuración que se usó para represntar un invernadero se necesitan definir primero las siguientes regiones:

$$\begin{aligned}
\delta\Omega_1 &= \{(X,Y) \in \Omega | X = 0 \vee X = 1 \vee \quad Y = 1\} \\
\delta\Omega_2 &= \{(X,Y) \in \Omega | (X < 0.500 \quad \vee X > 1.050) \wedge (Y = 0)\} \\
\delta\Omega_3 &= \{(X,Y) \in \Omega | (0.500 <= X <= 1.050 \wedge (Y = 0)\} \\
\delta\Omega_4 &= \{(X,Y) \in \Omega | (0.500 <= X <= 1.050) \wedge (0.489 < Y < 0.500)\} \\
\delta\Omega_5 &= \{(X,Y) \in \Omega | (0.500 <= X <= 0.530) \wedge (0 < Y < 0.500)\} \\
\delta\Omega_6 &= \{(X,Y) \in \Omega | (1.020 <= X <= 1.035) \wedge (0.0 < Y <= 0.250)\} \\
\delta\Omega_7 &= \{(X,Y) \in \Omega | (1.020 <= X <= 1.035) \wedge (0.350 =< Y < 0.500)\} \\
\delta\Omega_T &= \delta\Omega_1 \cup \delta\Omega_2 \cup \delta\Omega_3 \cup \ldots \delta\Omega_7
\end{aligned}$$

Así, las condiciones de frontera, para la componente X de la velocidad (U), la componente Y de la velocidad (V), la corrección a la presión P' (ver sección 2.3.3) y la temperatura θ se definen cómo:

$U = V = P' = 0$ para $\delta\Omega_T$
$\theta = T_a = 0$ para $\delta\Omega_1 \cup \delta\Omega_2$
$\theta = T_s = 1.0$ para $\delta\Omega_3$
$\theta = T_r = 0.439$ para $\delta\Omega_4$
$\frac{\partial\theta}{\partial X} = 0$ para $\delta\Omega_5 \cup \delta\Omega_6 \cup \delta\Omega_7$

La definición formal del resto de configuraciones es similar y se omite en esta sección. En la sección de reultados se muestran esquemas que representan las distintas configuraciones. Con estos esquemas es suficiente para dar una interpretación a los resultados obtenidos. Pasemos entonces al modelo numérico.

2.3. Modelo Numérico: Método del volumen finito

2.3.1. Introducción al FVM y discretización del espacio

Podemos describir de forma general al *Método de Volumen Finito* (FVM) como un método que parte de un modelo definido por ecuaciones diferenciales parciales sobre un domino físico para luego definir una malla discreta construida por volúmenes sobre dicho dominio, realizar la integración de las ecuaciones sobre cada volumen y así obtener un conjunto de ecuaciones lineales algebraicas discretas. En el caso de estudio que nos ocupa, la interpretación física de dichas ecuaciones algebraicas es que estas expresan, de forma discreta, el principio de conservación para la variable ϕ (que puede representar a la temperatura o a alguna componente de la velocidad) en cada volumen de control. El principio de conservación se expresa mediante una combinación lineal de los valores discretos de ϕ en cada volumen de control y en sus vecinos más cercanos.
Que este método esté basado en el principio de conservación es uno de los atractivos del FVM puesto que incluso para una malla gruesa, es decir con relativamente pocos volúmenes, la solución exhibirá un comportamiento conservativo, esto en términos simples significa quelo que entra debe salir. Podemos decir que el comportamiento conservativo de la variable ϕ se preserva independientemente del número de puntos de la malla y no sólo cuando la malla es excesivamente fina y esta es una de las principales razones por las se utilizó el método de volumen finito (FVM) como modelo numérico para resolver la ecuación general de transporte (ecuación 2.17). Dicha ecuación está definida sobre un espacio continuo y, como hemos dicho, el primer paso consiste en discretizar el espacio mediante la definición de una malla que segmente el dominio en N regiones conocidas como *volúmenes de control* (V_C). En la figura 2.2 se muesta el ejemplo para una malla con $N = 20$ volúmenes de control en un dominio cuadrado de dos dimensiones. La ecuación general de transporte es válida para todo el dominio y particular para cada uno de los

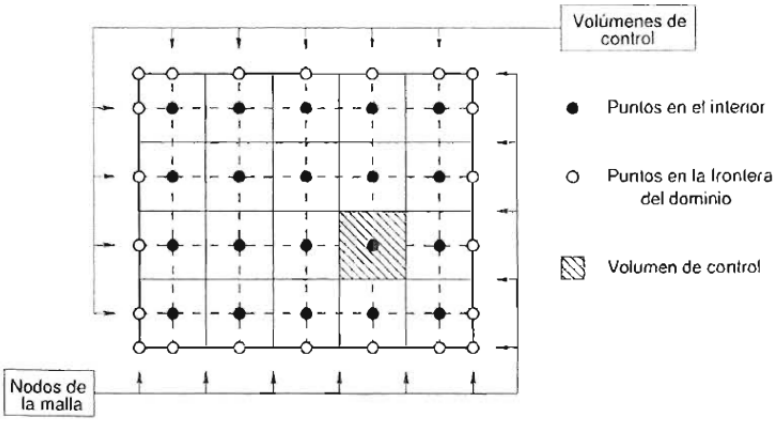


FIGURA 2.2: Dominio 2D discretizado usando una malla con 20 volúmenes de control.

volúmenes (o rectángulos) de control. Con el objetivo de mostrar el caso más general pensemos en el caso de 3 dimeniones. Si integramos la ecuación 2.17 sobre cada uno de los volúmenes de control se obtiene la siguiente expresión:

$$\int_{V_C} \boldsymbol{\nabla} \cdot (\vec{\mathbf{u}}(x,y,z)\phi(x,y,z) - \Gamma(x,y,z)\boldsymbol{\nabla}\phi(x,y,z))dV = \int_{V_C} S(x,y,z)\ dV \tag{2.18}$$

En la expresión anterior, como en todas las que siguen hasta la ecuación 2.22 las variables se representan con funciones continuas que dependen de la posición.
Por otro lado el teorema integral de Gauss (teorema de la divergencia) dice que para un campo vectorial $\vec{\mathbf{F}}$ de clase C^1 se tiene que:

$$\int_D \boldsymbol{\nabla} \cdot \vec{\mathbf{F}}(x,y,z)dV = \int_{\partial D} \vec{\mathbf{F}}(x,y,z) \cdot \vec{\mathbf{n}}dA \tag{2.19}$$

Donde D representa el volumen sobre el que se integra y ∂D su respectiva frontera. Usando este resultado en la ecuación 2.18 resulta:

$$\int_{\partial V_C} (\vec{\mathbf{u}}(x,y,z)\phi(x,y,z) - \Gamma(x,y,z)\boldsymbol{\nabla}\phi(x,y,z)) \cdot \vec{\mathbf{n}}dA = \int_{V_C} S\ dV \tag{2.20}$$

Donde ∂V_C representa la frontera del volumen de control. Por simplicidad, y porque así ocurre en la implementación del presente trabajo, de aquí en adelante supondremos que la malla es cartesiana por lo que todos los volúmenes de control son paralelepípedos (para los casos 3D) o rectángulos (para los casos 2D). Fijémonos en el caso 3D por ser el más general. Por consiguiente, la frontera ∂V_C está formada por 6 caras (ver figura 2.3). Para un volumen de control centrado en un nodo P se tienen 6 caras en las direcciones: $\hat{\imath}$, $-\hat{\imath}$, $\hat{\jmath}$, $-\hat{\jmath}$, $\hat{k}$, $-\hat{k}$, que se denotan como: e, w, n, s, f, b, respectivamente. De manera análoga los 6 nodos vecinos del nodo P en las direcciones: $\hat{\imath}$, $-\hat{\imath}$, $\hat{\jmath}$, $-\hat{\jmath}$, $\hat{k}$, $-\hat{k}$,

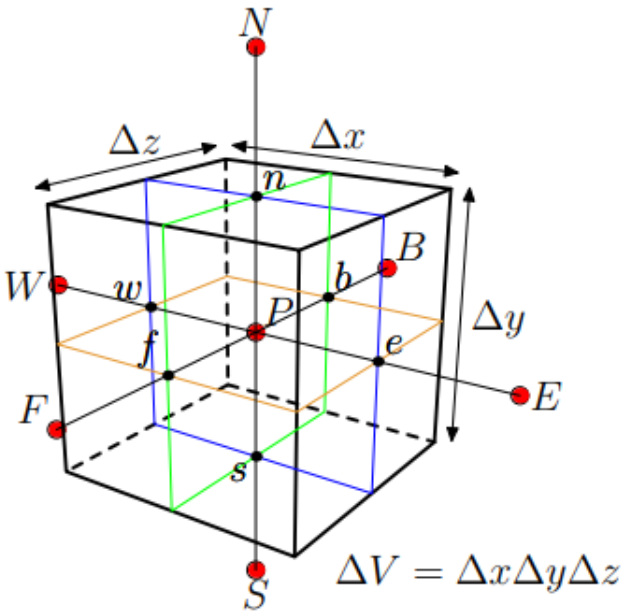


FIGURA 2.3: Volumen de control en tres dimensiones y nomenclatura de los vecinos.

se denotan como: E, W, N, S, F y B. La notación de las caras del volumen y los nodos vecinos se ilustra en la figura 2.3. De esta manera, la integral sobre la frontera del volumen de control (∂Vc) expresada en la ecuación 2.20 se puede separar como la suma de 6 integrales sobre cada una de las caras del volumen de control (e, w, n, s, f, b) como sigue:

$$\sum_{m}^{6}\left(\int_{cara_m}(\vec{\mathbf{u}}(x,y,z)\phi(x,y,z)-\Gamma(x,y,z)\boldsymbol{\nabla}\phi(x,y,z))\cdot\hat{\mathbf{n}}dA\right)=\int_{V_C}S(x,y,z)\ dV \tag{2.21}$$

$$\sum_{m}^{6}\int_{cara_m}\vec{\mathbf{u}}\phi(x,y,z)\cdot\hat{\mathbf{n}}dA-\sum_{m}^{6}\int_{cara_m}\Gamma\boldsymbol{\nabla}\phi(x,y,z)\cdot\hat{\mathbf{n}}dA=\int_{V_C}S(x,y,z)\ dV \tag{2.22}$$

Donde $cara_m = cara_1, cara_2, cara_3, cara_4, cara_5, cara_6 = e, w, n, s, f, b$.
Las integrales de la ecuación 2.22 son integrales sobre funciones continuas y, en principio desconocidas. Con el objetivo de discretizar dicha ecuación se hace la aproximación de considerar que todos los integrandos son constantes y, en el caso de las integrales sobre las caras, sólo dependen de la cara sobre la que se integra. Esta aproximación es similar a la *forma integral del teorema del valor medio* y mejora a medida que el tamaño del volumen de control tiende a cero, es decir, entre más nodos tenga la malla mejor será la aproximación. Con base en esta aproximación, de la ecuación 2.22 se obtiene la siguiente

expresión discreta para la ecuación de transporte:

$$\sum_{m}^{6} \phi_m A_m \vec{\mathbf{u}}_m \cdot \hat{\mathbf{n}}_m - \sum_{m}^{6} \Gamma_m A_m (\boldsymbol{\nabla}\phi)_m \cdot \hat{\mathbf{n}}_m = S_p V_{ol} \tag{2.23}$$

En esta expresión el subíndice de las variables ϕ,$\vec{\mathbf{u}}$,Γ y $\boldsymbol{\nabla}\phi$ se usa para indicar que dichas variables están evaluadas en la respectiva cara del volumen. Por ejemplo, $\phi_1 = \phi_e = \phi(x + \Delta x/2, y, z)$ El subíndice de $\hat{\mathbf{n}}$ indica que $\hat{\mathbf{n}}_m$=$\hat{\mathbf{n}}_1, \hat{\mathbf{n}}_2, \hat{\mathbf{n}}_3, \hat{\mathbf{n}}_4, \hat{\mathbf{n}}_5, \hat{\mathbf{n}}_6$=$\hat{\imath}, -\hat{\imath}, \hat{\jmath}, -\hat{\jmath}, \hat{k}, -\hat{k}$.

2.3.2. Aproximación de los términos advectivos y difusivos

Los términos de la suma de la izquierda en la ecuación2.23 (los que dependen de las velocidades) se conocen como los *términos advectivos* y los términos de la suma a la derecha (los que contienen los factores Γ) se conocen como *términos difusivos*. Comencemos por analizar los términos difusivos, al desarrollar y expandir los términos difusivos de la ecuación2.23 se obtiene:

$$\begin{aligned}
&\sum_{m}^{6} \Gamma_m A_m (\boldsymbol{\nabla}\phi)_m \cdot \hat{\mathbf{n}}_m = \\
&= \Gamma_e (\boldsymbol{\nabla}\phi)_e \cdot \hat{\imath} A_e - \Gamma_w (\boldsymbol{\nabla}\phi)_w \cdot \hat{\imath} A_w + \Gamma_n (\boldsymbol{\nabla}\phi)_n \cdot \hat{\jmath} A_n \\
&\quad - \Gamma_s (\boldsymbol{\nabla}\phi)_s \cdot \hat{\jmath} A_s + \Gamma_f (\boldsymbol{\nabla}\phi)_f \cdot \hat{k} A_f - \Gamma_b (\boldsymbol{\nabla}\phi)_b \cdot \hat{k} A_b \\
&= \Gamma_e \frac{\partial \phi}{\partial x}\Big|_e A_e - \Gamma_w \frac{\partial \phi}{\partial x}\Big|_w A_w + \Gamma_n \frac{\partial \phi}{\partial y}\Big|_n A_n \\
&\quad - \Gamma_s \frac{\partial \phi}{\partial y}\Big|_s A_s + \Gamma_f \frac{\partial \phi}{\partial z}\Big|_f A_f - \Gamma_b \frac{\partial \phi}{\partial z}\Big|_b A_b
\end{aligned} \tag{2.24}$$

Después, para calcular las derivadas parciales de ϕ sobre las fronteras se usan *diferencias centradas* sobre las caras de los volúmenes y así se logra expresar a los términos difusivos como una combinación lineal de los valores de ϕ sobre el nodo en cuestión y sus vecinos más cercanos:

$$\begin{aligned}
&\sum_{m}^{6} \Gamma_m A_m (\boldsymbol{\nabla}\phi)_m \cdot \hat{\mathbf{n}}_m = \\
&= D_P \phi_P + D_E \phi_E + D_W \phi_W + D_N \phi_N + D_S \phi_S + D_F \phi_F + D_B \phi_B
\end{aligned} \tag{2.25}$$

Donde $D_P = -\sum_{vecino} D_{vecino}$, así como $D_{vecino} = \frac{\Gamma_{cara} A_{cara}}{\Delta P_{vecino}}$ definiendo A_{cara} como el área de la cara respectiva y ΔP_{vecino} como la distancia del nodo P a su nodo *vecino*, con $cara = e, w, n, s, f, b$ y su rescpectivo $vecino = E, W, N, S, F, B$.

De manera similar, se busca expresar a los términos advectivos como una combinación lineal de los valores de ϕ sobre el centro del volumen de control y sobre los nodos vecinos. Recordando que la velocidad $\vec{\mathbf{u}}$ se puede expresar en términos de sus componentes como $\vec{\mathbf{u}} = u\hat{\imath} + v\hat{\jmath} + w\hat{k}$ al expandir los

términos advetivos de la ecuación2.23 resulta:

$$\begin{aligned}\sum_{m}^{6}\phi_m A_m \vec{\mathbf{u}}_m \cdot \hat{\mathbf{n}}_m = \\ = A_e\phi_e\vec{\mathbf{u}}_e \cdot \hat{\imath} - A_w\phi_w\vec{\mathbf{u}}_w \cdot \hat{\imath} + A_n\phi_n\vec{\mathbf{u}}_n \cdot \hat{\jmath} \\ -A_s\phi_s\vec{\mathbf{u}}_s \cdot \hat{\jmath} + A_f\phi_f\vec{\mathbf{u}}_f \cdot \hat{k} - A_b\phi_b\vec{\mathbf{u}}_b \cdot \hat{k} \\ = u_e A_e\phi_e - u_w A_w\phi_w + v_n A_n\phi_n \\ -v_s A_s\phi_s + u_f A_f\phi_f - u_b A_b\phi_b\end{aligned} \tag{2.26}$$

Como podemos observar la variable ϕ queda evaluada sobre la caras del volumen de control y no sobre los nodos vecinos. Para aproximar los valores de ϕ sobre las caras en términos de ϕ sobre lo nodos se utiliza uno de varios *esquemas*. En el *esquema upwind* (de primer orden) se considera que el valor de ϕ_{cara} es igual al valor de ϕ sobre el nodo de donde viene el flujo; esto es, si el flujo viene de izquierda a derecha entonces ϕ_{cara} es igual al valor de ϕ en el nodo de la izquierda y viceversa, si el flujo viene de derecha a izquierda entonces es igual al valor de ϕ sobre el nodo de la derecha. De esta manera, se logra en efecto expresar a los términos advectivos como una combinación lineal:

$$\begin{aligned}\sum_{m}^{6}\phi_m A_m \vec{\mathbf{u}}_m \cdot \hat{\mathbf{n}}_m = \\ = F_P\phi_P + F_E\phi_E + F_W\phi_W + F_N\phi_N + F_S\phi_S + F_F\phi_F + F_B\phi_B\end{aligned} \tag{2.27}$$

Al realizar las aproximaciones descritas para los términos advectivos y difusivos se obtiene entonces un sistema de ecuaciones lineales que se puede resolver para determinar los valores de la variable ϕ en cada uno de los puntos de la malla. De esta forma, en principio, bastaría con resolver un sistema de ecuaciones por cada variable que sea de interés en el sistema físico. Sin embargo, este procedmiento es factible solo si cada variable física de interés presenta su propia ecuación diferencial que es independiente del resto y desafortunadamente este no es el caso de la *convección natural*. En nuestro caso de estudio, aún con las simplificaciones propuestas, se tiene una dependencia entre las ecuaciones diferenciales y cuando se da esta condición se dice que las ecuaciones diferenciales están *acopladas*, adcionalmente las ecuaciones son no lineales. A continuación se describe una manera de desacoplar las ecuaciones diferenciales.

2.3.3. Desacoplamiento mediante el método SIMPLE

En el presente trabajo se debe tratar un modelo descrito por ecuaciones diferenciales acopladas como las siguientes:

$$\frac{\partial U}{\partial X} + \frac{\partial V}{\partial Y} = 0 \tag{2.28}$$

$$U\frac{\partial U}{\partial X} + V\frac{\partial U}{\partial Y} = -\frac{\partial P}{\partial X} + Pr\left[\frac{\partial^2 U}{\partial X^2} + \frac{\partial^2 U}{\partial Y^2}\right] \tag{2.29}$$

$$U\frac{\partial V}{\partial V} + V\frac{\partial V}{\partial Y} = -\frac{\partial P}{\partial Y} + Pr\left[\frac{\partial^2 V}{\partial X^2} + \frac{\partial^2 V}{\partial Y^2}\right] + PrRa\theta \tag{2.30}$$

$$U\frac{\partial \theta}{\partial X} + V\frac{\partial \theta}{\partial Y} = \left[\frac{\partial^2 \theta}{\partial X^2} + \frac{\partial^2 \theta}{\partial Y^2}\right] \tag{2.31}$$

Uno de los métodos para desacolpar las ecuaciones es conocido como *Semi-implicit method for pressure-linked equations*(SIMPLE) [16] y para simplificar la discusión de esta técnica nos referiremos a las ecuaciones 2.28-2.31 como ecuaciones de *continuidad, momento X, momento Y y temperatura*; respectivamente. Una vez que se aplica el método de volumen finito (secciones 2.3.1 y 2.3.2) a las ecuaciones de momento y temperatura se obtienen ecuaciones **discretas** de momento y temperatura. Además, la ecuación de continuidad puede expresarse mediante diferencias entre los valores discretos de U y V obtenidos del FVM. Cuando la versión discreta de la ecuación de continuidad se cumple se dice que la velocidad satisface la *condición de continuidad*. El método SIMPLE es un procedimiento que consiste, de forma básica, en resolver las ecuaciones de momento y temperatura de forma iterativa de tal manera que con el paso de las iteraciones se mejore la *condición de continuidad*. Esto se consigue resolviendo en cada iteración una ecuación para calcular una nueva presión P'; esta P' se usa para actualizar la presión P con la que se obtienen velocidades U y V que son mejores en términos de la *condición de continuidad*. A la ecuación discreta que se usa para calcular P' se le conoce como *ecuación de corrección a la presión*[16]. A continuación, se presenta de forma general los pasos del método SIMPLE:

1. Asignar la temperatura, presión y velocidades iniciales.
2. Resolver las ecuaciones discretizadas de momento a partir de los valores conocidos.
3. Resolver la ecuación de correción a la presión para encontrar la nueva variable P'.
4. Con los valores de correción a la presión P' se calculan nuevos valores de velocidad y presión.
5. Resolver el resto de ecuaciones de transporte, la ecuación de temperatura en este caso.
6. Si no se ha alcanzado la convergencia volver al paso 2, en caso contrario termina.

Además, es importante mencionar que las variables temperatura (T) y presión (p) se calculan sobre la misma malla mientras que las compomentes de la velocidad se calculan sobre mallas distintas conocidas como *mallas recorridas* (ver Figura 2.4).

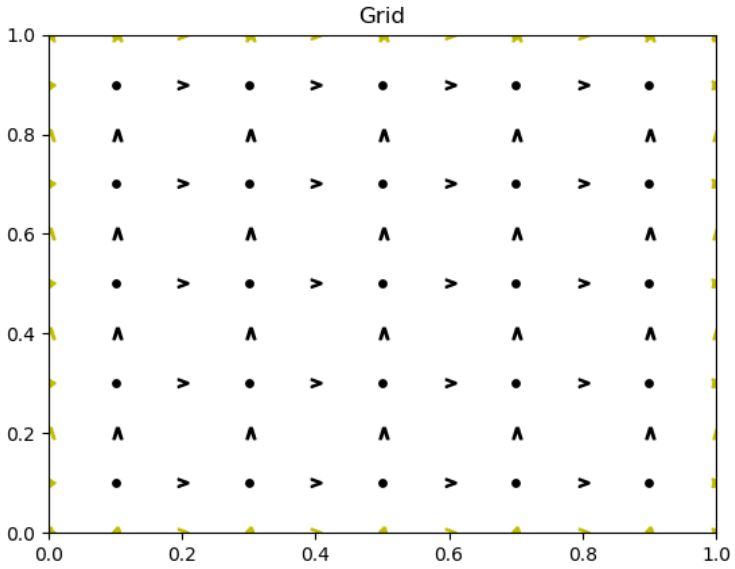


FIGURA 2.4: Mallas recorridas. La malla donde se calculan T y p está representada por los puntos • mientras que las mallas donde se calculan las componentes de la velocidad ,u y v, están representadas por $>$ y $\wedge$, respectivamente.

En atención a lo expuesto, se debe de resolver un sistema de ecuaciones lineales por cada ecuación diferencial del modelo, además esto debe de iterarse n veces hasta alcanzar una tolerancia definida en términos de la ecuación de continuidad de acuerdo con el método SIMPLE. Esto puede también requerir de repetirse m veces en el caso de un problema no-estacionario. Si además consideramos que los sistemas de ecuaciones pueden ser de miles o millones de incógnitas resulta claro que la resolución de sistemas de ecuaciones lineales es fundamental en la implementación del FVM. La utilización de sistemas de ecuaciones tan grandes obedece al hecho de que en el FVM se requiere de la previa construcción de una malla sobre el dominio de estudio y para obtener una simulación realista del fenómeno se requiere de una malla de suficiente precisión, es decir de un número alto de nodos. Por ejemplo, en [4], se usan del orden de $N = 10^5$ y $N = 10^6$ para obtener sus reultados (N es el número de volúmenes). El sistema de ecuaciones que genera el FVM está representado por una matriz de tamaño N^2 (pues resolvemos para N incógnitas) por lo que la implementación tiene 2 retos principales a resolver, el almacenamiento en memoria y el tiempo para resolver un sistema de ecuaciones con esas dimensiones. Debido a que las matrices que arroja el FVM son dispersas, el primer desafío (el almacenamiento de memoria) puede atacarse con un formato de almacenado de datos especializado para matrices

dispersas, como puede ser el caso del *compressed sparse row* (CSR)[17]. Con el propósito de resolver sistemas de ecuaciones es clave desarrollar una implementación que sea capaz de leer el formato CSR y resolver de forma eficiente dichos sistemas. Para la solución de sistemas de ecuaciones de gran tamaño es común el uso de métodos iterativos como GMRES.

2.3.4. Número de Nusselt

Una la formas para medir de forma cuantitativa la transferencia de calor es mediante el número de Nusselt (Nu). El número de Nusselt local, en la dirección X, se define cómo [14]:

$$Nu_{x,local}(X,Y,Z) = v_x\theta - \frac{\partial\theta}{\partial X} \tag{2.32}$$

Donde θ es la temperatura adimensional y v_x es la componente x de la velocidad (definiciones similares pueden hacerse en las direcciones 'Y' y 'Z'). De la definición puede verse que un Nu local negativo en la en la pared izquierda indica una transferencia de calor hacia afuera de cavidad y un Nu local positivo sobre la misma pared significaría que hay transferencia hacia adentro de la cavidad [14]. En general, un Nu local positivo indica que la transferencia es en el sentido positivo de las 'X'. Para obtener el Nu promedio se debe de integrar el Nu local sobre una región y dividir entre la medida de dicha región. Así, para calcular el Nu promedio sobre la pared izquierda de una cavidad cuadrada de largo igual a 1 se tiene:

$$Nu_{prom} = \int_0^1 Nu_{x,local}(0,Y)dy \tag{2.33}$$

Para una malla discreta como la del FVM la derivada de $Nu_{x,local}$ puede calcularse con una aproximación de segundo orden y la integral de Nu_{prom} se puede hacer de forma numérica con el método de Simpson.

2.3.5. Solución de sistemas lineales

Existen una gran cantidad de métodos para resolver sistemas de ecuaciones lineales, es común referirse a dichos métodos como *solvers*. Una posible clasificación de estos métodos los separa en dos grupos: los directos y los iterativos. Los *solvers* directos tienen un número definido de pasos que al llevarse a cabo se obtiene un único vector solución $\mathbf{x}$. En contraste, los métodos iterativos obtienen una sucesión de vectores $\mathbf{x}_i$ que representan soluciones aproximadas y no tiene un número de pasos definidos sino que se detiene una vez que la solución $\mathbf{x}_i$ es una aproximación que se considera suficientemente buena. Dos algoritmos típicos de los métodos directos e iterativos son la factorización *LU* y el método *Jacobi*, respectivamente.
Usar los métodos directos puede ser costoso computacionalmente en términos de tiempo y almacenamiento. El uso de métodos iterativos puede dar una significativa disminución del costo computacional y la complejidad [22].

No obstante, no cualquier método iterativo es adecuado para problemas representados por matrices muy extensas pues métodos como el de *Jacobi* es de orden cúbico $O(n^3)$ [21]. Los *solvers* conocidos como *métodos del subespacio de Krylov* han probado su eficiencia para atacar sistemas de ecuaciones lineales de gran tamaño. Dos de los *métodos de Krylov* más populares son el *Generalized minimal residual method* (GMRES) y el *Biconjugate Gradient Stabilized Method* (BICGSTAB)[17], en las siguientes secciones nos concentraremos en estos métodos.

GMRES

El GMRES es un método iterativo para resolver sistemas de ecuaciones lineales que tiene la propiedad de minimizar, en cada paso, la norma del vector residuo sobre un subespacio de Krylov. El algoritmo se deriva de el proceso de Arnoldi para constuir una base ortogonal l_2 del subespacio de Krylov. GMRES puede ser considerado como una generalización del algoritmo MINRES [20]. El algoritmo parte de una estimación inicial x_0 y, de acuerdo con [21], se puede enunciar de la siguiente manera:

1. Se calculan $r_0 = b - Ax_0$, $\beta := ||r_0||_2$ y $v_1 := r_0/\beta$
2. For j=1,2,3,...,m:
   a) Computar $\omega_j = Av_j$
   b) For i=1,...,j:
      1) $h_{ij} = (\omega_j, v_i)$
      2) $\omega_j = h_{ij}v_j$
   c) Termina *For*
   d) $h_{j+1,j} = ||w_j||_2$
   e) Si $h_{j+1,j} = 0$ define $m = j$ y termina *For*
   f) $v_{j+1} = \omega_j/h_{j+1,j}$
3. Termina *For*
4. Define la $(m+1) * m$ matriz de Hessenberg $H_m = \{h_{ij}\}_{1\leq i\leq m+1,\quad 1\leq j\leq m}$
5. Se calcula y_m que minimiza $||\beta e_1 - H_m y_m||_2$ y $x_m = x_0 + V_m y_m$ usando el método de mínimos cuadrados

BICGSTAB

A continuación se muestra el algoritmo iterativo de BICGSTAB para resolver el sistema de ecuaciones lineales $\mathbf{Ax} = \mathbf{b}$ [22]:

1. Se define la suposición inicial $\mathbf{x}_0$ (arbitraria).
2. $\mathbf{r}_0 = \mathbf{b} - \mathbf{Ax}_0$
3. Se elige un $\mathbf{r}_0^*$ arbitrario de tal manera que $< \mathbf{r}_0, \mathbf{r}_0^* > \neq 0$

4. Se define $\mathbf{p}_0 = \mathbf{r}_0$ y $\omega_0 = 1$
5. for $i = 1, 2, 3, ...$
   a) $\rho = < \mathbf{r}_i, \mathbf{r}_0^* >$
   b) $\mathbf{u}_i = \mathbf{A}\mathbf{p}_i$
   c) $\alpha_i = \frac{\rho}{<\mathbf{u}_i, \mathbf{r}_0^*>}$
   d) $\mathbf{s}_i = \mathbf{r}_i - \alpha_i \mathbf{u}_i$
   e) $\mathbf{t} = \mathbf{A}\mathbf{s}_i$
   f) $\omega_{i+i} = \frac{<\mathbf{t}, \mathbf{s}_i>}{<\mathbf{t}, \mathbf{t}>}$
   g) $\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{p}_i + \omega_i \mathbf{s}_i$
   h) Si $\mathbf{x}_{i+1}$ es lo suficientemente preciso entonces termina.
   i) $r_{i+1} = \mathbf{s}_i - \omega_i \mathbf{t}$
   j) $\beta_i = \frac{<\mathbf{r}_{i+1}, \mathbf{r}_0^*> \alpha_i}{\rho \omega_i}$
   k) $\mathbf{p}_{i+1} = \mathbf{r}_{i+1} + \beta_i (\mathbf{p}_i - \omega_i \mathbf{u}_i)$

Donde la tipografía en negritas denota un vector o matriz y $< \mathbf{a}, \mathbf{b} >$ denota el producto escalar de los vectores **a** y **b**.

Precondicionadores

El precondicionamiento es una técnica para mejorar el número de condición de una matriz. Esto puede suceder, por ejemplo, porque la matriz es demasiado extensa. La idea escencial de un precondicionador es muy simple. Supongamos el sistema de ecuaciones lineales:

$$Ax = b \tag{2.34}$$

Es decir:

$$AIx = b \tag{2.35}$$

Donde I es la matriz identidad, si suponemos que se tiene una matriz M invertible, tendríamos equivalentemente:

$$AM^{-1}Mx = b \tag{2.36}$$

Si definimos $y = Mx$, $D = AM^{-1}$ la expresión anterior (ecuación2.36) nos queda:

$$Dy = b \tag{2.37}$$

Que es un sistema de ecuaciones definido por la matriz D que se puede resolver para encontrar el vector solución y. Después el vector x se encuentra haciendo $x = M^{-1}y$. La idea es que resolver el sistema de ecuaciones definido por ecuación2.37 es más simple que resolver el sistema definido por ecuación2.34. A la matriz M se le conoce como precondicionador y a este

procedimiento se le conoce como precondicionamiento por la derecha. Alternativamente se puede precondicionar por la izquierda si tomamos el sistema original (ecuación2.34) y multiplicamos M^{-1} por la izquierda para obtener:

$$M^{-1}Ax = M^{-1}b \tag{2.38}$$

Es decir:

$$(M^{-1}A)x = c \tag{2.39}$$

Donde $c = M^{-1}b$ y la matriz $M^{-1}A$ tiene un número de condición más bajo. Cabe destacar que en el procedimiento de precondicionamiento es necesario calcular primero la inversa del precondicionador (M^{-1}) e idealmente este debe de ser un proceso lo suficientemente simple. En efecto, un buen precondicionador debe cuplir los siguientes requerimientos:

- El sistema precondicionado debe ser fácil de resolver.
- El precondicionador debe ser simple de construir y aplicar.

Sin embargo, estos dos requirimientos están en competencia el uno con el otro, es decir, para obtener un sistema precondicionado se debe de considerar el intercambio que existe entre estas dos propiedades.

Capítulo 3

Desarrollo del Software y primeras pruebas

En el capítulo anterior se describe el problema a resolver, el marco en el que está definido y el método numérico que se eligió para resolverlo. En este capítulo se describe la forma en que fué implementado dicho método así como algunos puntos claves de la implementación. Además, en la segunda parte de este tercer capítulo (sección 3.3) se muestran los primeros ensayos en problemas de convección natural. Estos ensayos representan un puente entre el desarrollo y los resultados pues las soluciones se pueden interpretar como resultados preliminares o que ayudan a sustentar los métodos desarrollados para la convección natural. Comencemos entonces, por discutir brevemente algunas consideraciones y caracterísiticas de la implementación.

3.1. Particularidades del software implementado

Existen varias particularidades del software desarrollado que cabe destacar, para empezar, discutiremos un poco del lenguaje de programación empleado para el desarrollo del software.

Python

En el cómputo científico es común la utilización de lenguajes de programación eficientes para la tarea realizada. Así pues, resulta típico el uso de lenguajes como *Fortran*, *C* o *C++* pues son lenguajes compilados que ofrecen un buen rendimiento. Sin embargo, se tomó la desición de desarrollar la implementación del presente trabajo en el lenguaje de programación *Python* debido a que el lenguaje cuenta con ventajas en los siguientes puntos:

1. Velocidad desarrollo de software
2. Paradigma orientado a objetos
3. Extensa comunidad
4. Colaboración y reutilización
5. Estudio de las capacidades del lenguaje

6. Existen varias implementaciones de Python que pueden optimizar el código

El primer punto señala que con *Python* se pueden producir implementaciones en menos tiempo que otros lenguajes. Esto es, por un lado, porque *Python* se considera un lenguaje de alto nivel lo que se traduce en un sencillo manejo y rápida curva de aprendizaje; además este lenguaje cuenta con una gran cantidad de bibliotecas (baterías incluidas) que aceleran el proceso de desarrollo. El segundo punto se refiere a que una de las características que fueron establecidas desde etapas muy tempranas del trabajo fue la de emplear un paradigma de programación orientado a objetos y *Python* es uno de los lenguajes que ofrecen esta oportunidad. La tercer propiedad se refiere a que *Python* es uno de los lenguajes más populares al momento de crear el software [23]. En consecuencia, se tienen disponibles una considerable cantidad de módulos que se han realizado para una variedad de aplicaciones y en particular para el cómputo científico. Además, la creciente comunidad del lenguaje brinda una documentación abundante que incluye libros, tutoriales, y foros de ayuda. En cuarto lugar, un lenguaje de programación tan popular facilita que otras personas puedan colaborar o reutilizar el código desarrollado en el presente trabajo y esto consiste uno de los principales objetivos del mismo. Otra de las metas se menciona en el penúltimo punto; si bien no existe un lenguaje de programación que sea ideal para cualquier situación, utilizar *Python* sirve para explorar las capacidades y limitaciones del lenguaje en el contexto del problema planteado. Finalmente, existen varias implementaciones de Python, la más usada conocida como CPython, basada en el lenguaje C y por ello es la más eficiente. Adicionalmente, es posible hacer uso de herramientas que permiten compilar código Python en tiempo de ejecución, conocidas como JIT (compilador Just In Time) y esto permite acelerar algunos códigos numéricos [24].

Funcionalidad en 1,2 y 3 dimensiones

Una de las misiones del código creado es la capacidad de resolver una gran variedad problemas. En particular problemas definidos en 1, 2 o 3 dimensiones. Para esto se pueden crear métodos que varíen de forma dependiendo de la dimension pero, en general, la manera en la que se trató este punto es definiendo métodos o atributos de la misma forma pero que sean lo suficientemente generales para realizar la tarea independientemente del número de dimensiones del problema. Por ejemplo, la implementación almacena los coeficientes que definen el sistema de ecuaciones del FVM en arreglos tridimensionales sin importar si se trata de un problema en 1, 2 o 3 dimensiones. Si una de las dimensiones no se usa entonces en número de volúmenes en esa dirección es igual a uno; así un dominio unidimensional almacena sus coeficientes en un arreglo de forma $(1, 1, n)$, donde n es el número de volúmenes. De esta manera, los elementos de los arreglos *siempre* pueden extraerse con tres índices k,j,i, de nuevo, independientemente del número de dimensiones. Esta filosofía fue usada para implementar métodos en la medida de lo posible. Así mismo, la malla también se construye siempre como un

prisma rectangular por lo que todos los nodos que no tienen sentido para la correspondiente dimensión son etiquetados con la cadena *'Off'*. Esto, junto con una correcta separación de estos nodos evita que interactúen con el resto del proceso.

Por otro lado, *Numpy* tiene sobrecargadas las operaciones de suma, resta, multiplicación y división (entre otras) para ser realizadas elemento a elemento. El cálculo de los coeficientes del FVM realiza este tipo de operaciones elemento a elemento y en la mayoría de tareas de esta natutaleza se usan estas operaciones sobrecargadas. Esto presenta dos ventajas; primeramente, permite un código más legible (arreglo1+arreglo2 en lugar de un triple, doble o simple for) y en segundo lugar, proporciona ventajas de desempeño dado que estas operaciones están implementadas de forma eficiente en *Numpy* y en algunos casos esto se conecta con bibliotecas de alto desempeño como BLAS y LAPACK.

Condiciones de frontera dentro del dominio

Para el problema que se ha planteado, es necesario definir condiciones de frontera dentro del dominio físico del modelo. Por ejemplo, el techo del invernadero se localiza cerca del centro del dominio y tiene definida una temperatura constante diferente a la temperatura constante del ambiente. De esta manera se tiene una condición de tipo Dirichlet pero que tiene puntos a su alrededor para los cuales debe calcularse la temperatura. Esto tiene dos complicaciones escencialmente; la primera es cómo se etiquetan los nodos correspondientes de acuerdo al modelo geométrico y la segunda cómo se omiten estos nodos en el sistema de ecuaciones, pues son nodos con temperatura conocida.

Respecto al primer punto, se tienen dos métodos que se encargan de eso. Por un lado, el método *setUsrTag()* asigna la etiqueta (cadena) a todos los nodos dentro del prisma rectangular definido por las coordenadas de sus esquinas. Adicionalmente, el método *tagByCond()* toma una función de las coordenadas $f(x, y, z)$ que devuelve una condición Booleana. Esta función le sirve al método para discriminar qué nodos deben de ser etiquetados. El método *tagByCond()* resulta muy útil para etiquetar, por ejemplo, los puntos dentro de un anillo, la única desventaja es que la función del anillo debe ser traducida a código por el usuario manualmente.

Como habíamos dicho, la segunda complicación de definir condiciones de frontera en el interior del dominio es la de crear un sistema de ecuaciones que no contenga las ecuaciones de los nodos que corresponden a estas fronteras. Este detalle se resuelve al crear ecuaciones que sean linealmente independientes al resto de ecuaciones del sistema. En efecto, si la malla tiene N volúmenes entonces el sistema de ecuaciones es de $N \times N$, la ecuación n-ésima es la ecuación discreta del volúmen número n donde se calcula la incógnita x_n y esta ecuación puede volverse independiente al resto de ecuaciones si se define como $x_n = b_n$. Donde b_n es igual a cero o cualquier término

independiente. Una vez resuelto el sistema de ecuaciones, se invoca un método creado para corregir el valor del x_n tomando en cuenta el tipo de frontera que se tiene definida en el nodo correspondiente.

Otro pormenor que vale la pena mencionar es el ahorro de memoria en el almacenamiento de las coordenadas de las mallas. Dado que las coordenadas de los nodos se forman de todas las combinaciones (x_i, y_j, z_k) con $i \in \{1, 2, \ldots, N_x\}$,$j \in \{1, 2, \ldots, N_y\}$,$k \in \{1, 2, \ldots, N_z\}$ entonces sólo se necesita guardar las listas de x_i's, y_j's y z_k's. Así se hace en la implementación y si es necesario utilizar las coordenadas el código genera las combinaciones en arreglos provisionales que son desechados después del cálculo. La implementación también cuenta con métodos capaces de manejar matrices en formato CSR como otra forma para que el software guarde memoria.

Por último, uno de los objetivos de este trabajo es que el software generado sea flexible, en el sentido de que sea capaz de resolver problemas de distintas características. Por ejemplo, el software debe ser apto para problemas en una, dos, o tres dimensiones. Es por esta idea de generalidad en el uso que se tomó la decisión de emplear un paradigma de programación orientado a objetos. En total se contruyeron 9 clases: *Mesh, Coefficients, EqSystem, Solplot, Diffusion, Advection, UMesh, VMesh, WMesh*. Sin embargo, antes de profundizar en las clases implementadas, resulta conveniente dar una visión general de la manera en que trabaja el software.

3.2. Modelo Computacional: Construcción del software

Dicho de una forma básica, la funcionalidad principal del software es la de tomar la información que define el problema (tamaño del dominio, condiciones de frontera, etcétera) y con esto definir una malla que sirve para construir un sistema de ecuaciones del que se obtiene la solución que representa la situación física. En el esquema de la figura 3.1, se ilustra a grandes rasgos los procesos; con sus entradas y salidas, que atraviesa la información en el software para llegar al resultado final. En dicha figura 3.1, se muestra una línea punteada que indica un procedimiento recursivo; mismo que corresponde al método SIMPLE (ver sección 2.3.3). En los problemas en los que el campo de velocidades es conocido, el flujo de información es lineal y no atraviesa la línea punteada. Ahora bien, recordemos que el método de volúmen finito consiste, de la forma más básica, en discretizar un espacio físico utilizando una malla de puntos para los cuales se expresa una ecuación diferencial como una relación algebraica entre vecinos y que en su conjunto forman un sistema de ecuaciones lineales. Las clases construidas, tienen la intención de seguir esta abstracción de la manera más directa posible (ver figura 3.2).

Para dar una idea general de cómo está articulado (en cuanto a clases) el software, a continuación se describe brevemente el procedimiento que se sigue al usar la implementación; apartando de la discusión la parte física del

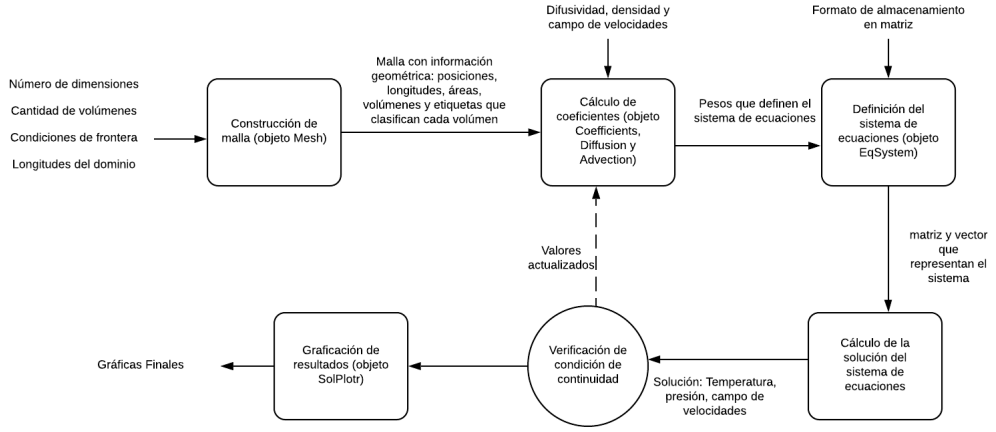


FIGURA 3.1: Esquema de la funcionalidad general del software. El diagrama representa la información que entra y sale en las distintas etapas de procesamiento.

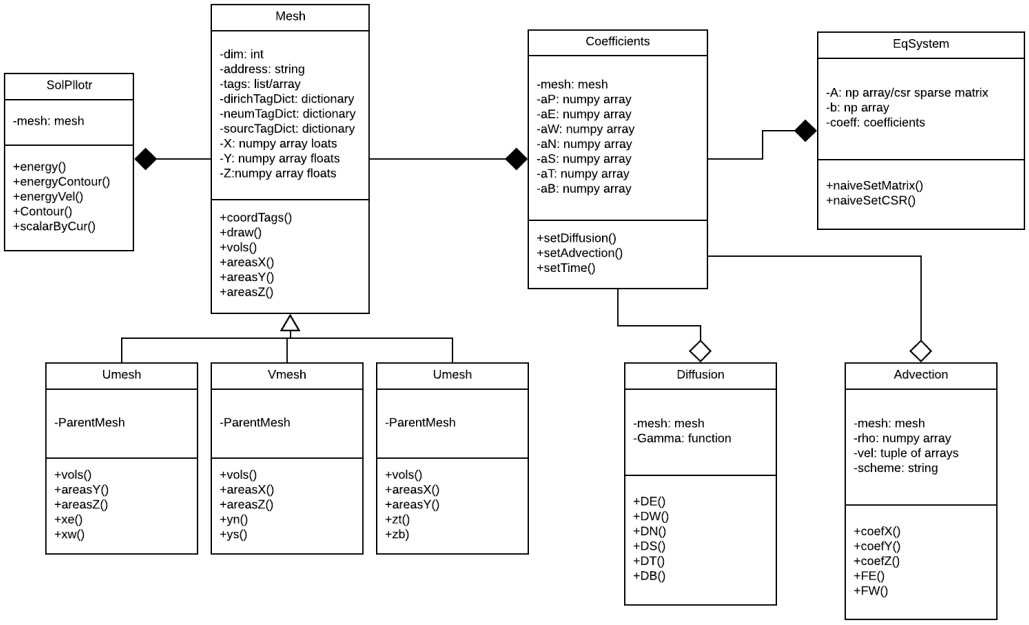


FIGURA 3.2: Diagrama de clases de clases para la implementación desarrollada del FVM.

problema y enfoncándonos en las clases y sus interacciones. Primeramente se crea un objeto de la clase *Mesh* que contiene la información geométrica y que define las fronteras físicas mediante los métodos de dicha clase. Luego, este objeto es utilizado para crear un objeto de la clase *Coefficients*. Posteriormente, se emplean los métodos de este objeto (*Coefficients*) para calcular la aportación *Difusiva, Advectiva* y/o *Temporal* del fenómeno. Para esto, la clase

Coefficients internamente se vale de objetos tipo *Difussion* y *Advection*. Después, usando el objeto tipo *Coefficients* se crea un objeto de la clase *EqSystem* para definir el sistema de ecuaciones lineales algebraicas que representa el fenómeno físico. Las soluciones de este sistema pueden graficarse entonces con ayuda de la clase *SolPlotr*.
Algunos problemas pueden requerir de varias mallas, sistemas de ecuaciones o iteraciones pero este es, de manera general, el funcionamiento del software.

3.2.1. Descripción de clases implementadas

Como hemos dicho, en su totalidad, la implementación desarrollada consta de 9 clases; cada una con una misión principal, tareas relevantes y forma particular de instanciar objetos. En las secciones siguientes, se describen brevemente algunos de estos puntos de las clases construidas.

Mesh

La clase *Mesh* es el corazón de la implementación pues el resto de clases tienen alguna conexión con esta. Esto se debe principalmente a que la clase *Mesh* maneja toda la infomación geométrica; qué coordenadas tienen los puntos de la malla, qué distancias hay entre puntos vecinos, qué puntos están en la frontera, etcétera. Los objetos de esta clase representan la malla de puntos que determinan los volúmenes en que se divide el espacio físico. Para instanciar un objeto de este tipo se debe indicar el número de dimensiones (1, 2 o 3) en las que está definido el problema teniendo en cuenta que el problema debe estar delimitado por un domino cartesiano; es decir, un segmento de recta, un rectángulo o un prisma rectangular. Tres de los atributos más importantes de un objeto tipo *Mesh* son los que definen las posiciones de los puntos de las mallas: *dominoX*, *dominoY* y *dominioZ*. Cada uno de estos atributos es una lista (arreglo de numpy) de números flotantes en orden ascendente que comienzan en 0, tales que las coordenadas de los puntos de la malla serían las combinaciones (x, y, z) formadas de elementos de $dominioX \times dominioY \times dominioZ$. Estos atributos pueden ser almacenados entregando los arreglos de forma directa a la clase o permitiendo que la clase los genere de forma uniforme al indicar las longitudes en cada dirección (dirección X, Y o Z) y el número de volúmenes que se tendrán en dicha dirección.
Para definir por completo el domino físico del problema es necesario además indicar qué fronteras y/o fuentes tiene el problema y cuáles son sus valores. Esta característica se implementa mediante el atributo *tags* que consiste en una lista de etiquetas (strings) que diferencían un punto de otro. Esto es, por cada punto de la malla se almacena una etiqueta significando que los puntos con la misma etiqueta pertenecen a un mismo grupo. Inicialmente, todas las etiquetas son *'I'* indicando que se trata de puntos *interiores* pero esta clase (*Mesh*) también incluye varios métodos construidos para modificar las etiquetas de acuerdo al problema planteado. Las etiquetas que comienzan con la letra *D* como 'D1' o 'D2' representan fronteras tipo Dirichlet, de igual manera la etiquetas que comienzan con *N* son fronteras tipo Neumann.

El número flotante que representa el valor de la variable o el flujo de la variable, en el caso Dirichlet y Neumann respectivamente, se almacena dentro de un diccionario para el que la llave al valor es la etiqueta correspondiente. Otro elemento clave en la construcción del software fue la implementación de un método que produjera una representación visual de la malla. Se trata del método *draw()* y resulta muy conveniente para verificar que la definición de la malla es correcta. Como se puede ver esta clase maneja toda la información del espacio físico y en consecuencia muchos de los métodos en las demás clases emplean un objeto tipo *Mesh*.

Coefficients

La clase *Coefficients*, por su parte, se usa para crear objetos que guardan y tratan todos los pesos (o coeficientes) que son necesarios para representar la ecuación diferencial de forma discreta. En efecto, de acuerdo con el método de volumen finito, cada volumen tiene una ecuación lineal algebraica asociada y los pesos que definen estas ecuaciones se guardan en los atributos de un objeto tipo *Coefficients*. Para continuar con la discusión de dichos atributos resulta útil explicar la nomenclatura adoptada y para ello, supongamos que la ecuación lineal algebraica de cada volumen se representa de la siguiente manera:

$$a_P\phi_P + a_E\phi_E + a_W\phi_W + a_N\phi_N + a_S\phi_S + a_T\phi_T + a_B\phi_B = S_u \qquad (3.1)$$

Donde los ϕ_{indice} representan las incógnitas, S_u es el término independiente y a_{indice} los coeficientes de acuerdo con el método de volumen finito. De modo que existe una ecuación como la anterior para cada uno de los volúmenes y por consiguiente existen tantos coeficientes a_P como volúmenes tiene la malla. Lo mismo ocurre para el resto de coeficientes a_{indice} y los términos independientes S_u. Volviendo a la implementación, como hemos dicho, una de las intenciones iniciales en su desarrolllo fue la de conseguir una abstracción casi directa del método (FVM). Es por esto que resultó natural definir los atributos; a_P, a_E, a_W, etcétera, como atributos de esta clase *Coefficients*. Estos atributos consisten en arreglos tridimensionales de números flotantes inicializados a cero y de hecho, para problemas de dimensiones menores a tres a algunos de estos atributos se les asigna el valor *None*. Es por esto que para generar un objeto de esta clase, se debe de proporcionar un objeto *Mesh* y dependiendo de este último se asigna la forma y la cantidad de memoria que tendrán los atributos del primero (ver tabla 3.1).

La idea principal de esta clase es crear un objeto asociado a la malla, que actualiza sus atributos dependiendo de la situación física a resolver. Así, para un modelo matemático que considere la difusión se construyó un método de la clase *Coefficients* llamado *setDiffusion()* que actualiza los pesos guardados en los atributos de manera correspondiente. Para el caso de modelos con advección se implementó el método *setAdvection()*. Estos métodos existen para plantear el modelo matemático de forma cómoda y natural; internamente, lo que hacen dichos métodos es manejar de forma automática objetos tipo *Diffusion* y *Advection*.

Dim	nvx	nvy	nvz	a_E	a_N	a_T
1	11	-	-	array(1,1,11)	None	None
2	15	15	-	array(1,15,15)	array(1,15,15)	None
3	8	9	6	array(6,9,8)	array(6,9,8)	array(6,9,8)

TABLA 3.1: Ejemplos de cómo quedan definidos los atributos a_E, a_N y a_T en función de los atributos de la malla dada. *Dim* es el número de dimensiones de la malla mientras que nvx, nvy y nvz es el número de volúmenes de la misma en las direcciones X,Y y Z, respectivamente. En estas columnas el símbolo $-$ significa que se tiene el valor por default (igual a 1). Además, *array()* indica que el atributo correspondiente es un arreglo de la forma definida por los números dentro del paréntesis.

Diffusion y Advection

Cuando un objeto *Coefficients* llama al método *setDiffusion()*, se crea inherentemente un objeto *Diffusion*. Este tipo de objeto contiene todos los métodos que permiten actualizar de forma transparente y automática los atributos del objeto *Coefficients* considerando la parte difusiva del problema. Estos cálculos requieren del coeficiente de difusividad (κ) y de la información geométrica de la malla. Por lo tanto, para generar un objeto *Diffusion* se deben dar los parámetros *mesh* y κ; donde *mesh* es un objeto tipo *Mesh* y κ es un flotante o función. La κ debe ser una función de las coordenadas ($f(x, y, z)$) en el caso de una κ que varía con la posición o un flotante si es constante para todo el dominio.
En cuanto a los objetos de la clase *Advection*, se trata de una situación análoga a la clase *Diffusion* pero estos tratan la parte advectiva del problema. En este caso, el método *setAdvection()*, que está implementado en la clase *Coefficients* genera automáticamente un objeto *Advection*. Para instanciar este objeto se requiere de los parámetros *mesh, rho, vel* y *scheme* que significan: un objeto *Mesh*, un número flotante con el valor de la densidad, una tupla con las tres componentes de la velocidad almacenada en tres arreglos y una cadena que indica el esquema advectivo, respectivamente.
Los objetos *Diffusion* y *Advection* obtienen la malla porque esta se encuentra almacenada también en el objeto *Coefficients* encargado de crearlos.

EqSystem

Por otro lado, la clase *EqSystem* toma un objeto tipo *Coefficients* y con este crea un sistema de ecuaciones; más específicamente, construye la matriz A y el vector b que representan un sistema de ecuaciones de la forma: $Ax = b$. Esto representa un nivel de abstracción en sí mismo pues la información de los coeficientes viene dada en forma de varios arreglos tridimensionales, y esta tiene que ser reinterpretada como elementos (renglón columna) de matriz. Además, el almacenamiento de la matriz puede ser completo (con todos los ceros) o utilizando algún formato de matriz dispersa, como el CSR o CSC.

Para instanciar un objeto *EqSystem* solo se requiere de un parámetro; el correspondiente objeto *Coefficients*. Luego el objeto *EqSystem* que se ha creado construye el sistema de ecuacioens usando alguno de los métodos de su clase. Se tienen varios métodos implementados que varían de acuerdo al formato de matriz seleccionado (FULL, CSC,CSR) y, de forma interna, a la dimensión de la malla (1,2 o 3). El sistema formado por la matriz A y el vector b, parámetros del objeto, puede entonces ser resuelto por un método directo o iterativo y la solución visualizada con la ayuda de la clase *SolPlotr*.

SolPlotr

El único fin de la clase *SolPlotr* es el de usar la información de la malla para asistir en la graficación de las soluciones. El parámetro requerido para instanciar un objeto de esta clase es un objeto tipo *Mesh*. La clase *SolPlotr* tiene implementados varios métodos para graficar una solución (almacenada en un arreglo) asociada a una malla (objeto *Mesh*). Dentro de estos métodos se encuentran gráficas con la solución representada por una escala de color, gráficas tipo campo de velocidades, gráficas de contorno, etc. Los métodos que obtienen gráficas para soluciones en 3D están basados en gráficas como las anteriores para tres cortes, es decir planos, localizados a un $'x'$, $'y'$ y $'z'$ constantes.

UMesh, VMesh y WMesh

Con respecto a las clases *UMesh, VMesh y WMesh* , tenemos que estas fueron diseñadas con el objetivo de tratar problemas de convección natural. En este tipo de problemas el campo de velocidades debe ser encontrado como parte de la solución al problema planteado y para ello se utilizan mallas ligeramente desplazadas con respecto a la original conocidas como *mallas recorridas*. Para una velocidad en tres dimensiones $\boldsymbol{\nu} = (U, V, W)$ se definen tres mallas desplazadas en los sentidos X,Y y Z que fueron implementadas al software mediante las clases *UMesh*,*VMesh* y *WMesh*; respectivamente.

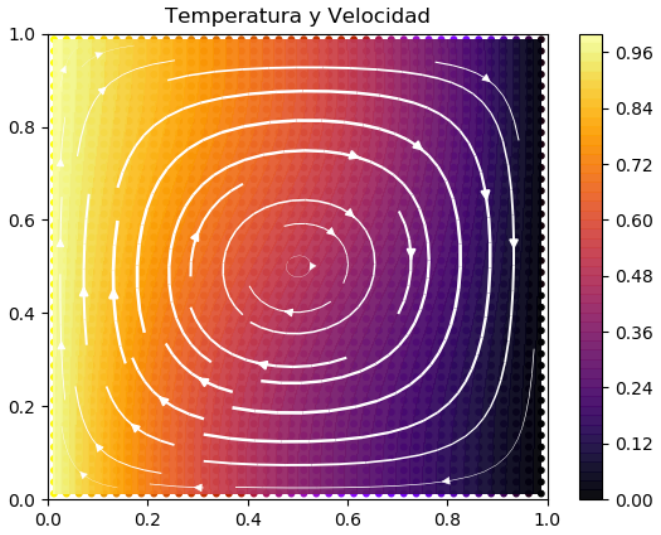
En general podemos decir que los trabajos citados en los *Antecedentes* (sección1.2) de la modelación térmica de invernadero utilizan un software previamente desarrollado o comercial (ANSYS/FLUENT en la mayoría de los casos) y es por eso que la discusión acerca del reto computacional es limitada. Dentro de este marco, se perciben relevantes los resultados de la implementación de FVM desarrollada para este trabajo de tesis. Sin embargo, para realizar un simulador fiable, es necesario verificar la congruencia de los resultados del simulador con valores probados. Por consiguiente, parte de la metodología de la presente tesis consistió en reproducir los resultados de trabajos previos en la solución del problema planteado.

3.3. Pruebas de validez de resultados a través de modelos conocidos

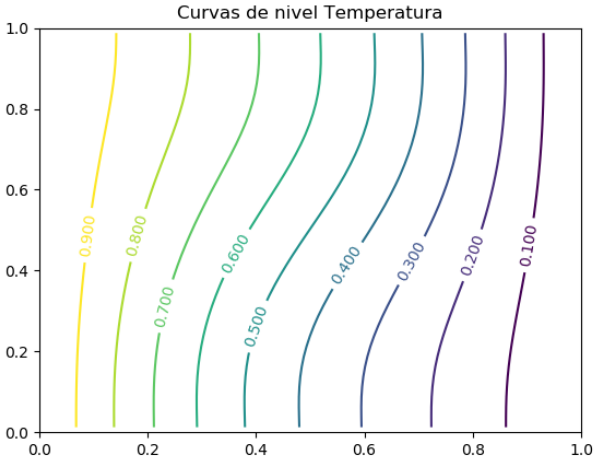
Para el presente trabajo, se consideraron algunos artículos académicos orientados al problema de la *convección natural* como un buen punto de partida para la validación de resultados, lo cual es relevante en la construcción de un software orientado a la computación científica. En el presente apartado, se exponen modelos conceptuales extraídos de tales trabajos y los resultados de su respectiva solución. En total se realizaron 4 pruebas (reproducciones).

3.3.1. Primer ejemplo de prueba

El primer texto académico que se utilizó como base para validar las soluciones del software implementado al problema de convección natural es *Natural Convection of Air in a Square Cavity a Bench Mark Numerical Solution*[11]. En dicho trabajo, se resuelven las ecuaciones adimensionales que describen la convección natural estacionaria en un dominio cuadrado cuando sus paredes laterales se someten a temperaturas constantes (T_h y T_c) y sus paredes superior e inferior son adiabáticas. Para estas condiciones de frontera, y utilizando la aproximación de Boussinesq, el texto toma un número de Prandtl (Pr) igual a 0.71 y define cuatro casos utilizando distintos números de Rayleigh (Ra). Son estos los resultados que se tomaron como base en la primera prueba de la implementación, (*Prueba 1*).
En la figura 3.3 se muestran los resultados de la reproducción de las solución para $Ra = 10^3$. En esta figura se puede observar que los resultados del campo de velocidades 3.3a y temperatura 3.3b son congruentes dado que las gráficas obtenidas y la teóricas [11] son muy similares. Esto es desde el punto de vista cualitativo; sin embargo, con el objetivo de realizar comparaciones cuantitativas se calculó también el número de Nusselt(Nu).
Más específicamente, el número de Nusselt promedio en la frontera vertical de la cavidad a $X = 0$ y el número de Nusselt promedio en la mitad de la cavidad ($X = 1/2$). El valor obtenido para estos números fue de 1.098 y 1.079; respectivamente, lo que representa una diferencia de 1.61 % y 3.31 % con respecto a los valores reportados en la referencia tomada [11]. El signo positivo de estos Nu promedio es consecuencia de que el flujo de temperatura en la cavidad es mayormente de izquierda a derecha.



(A)



(B)

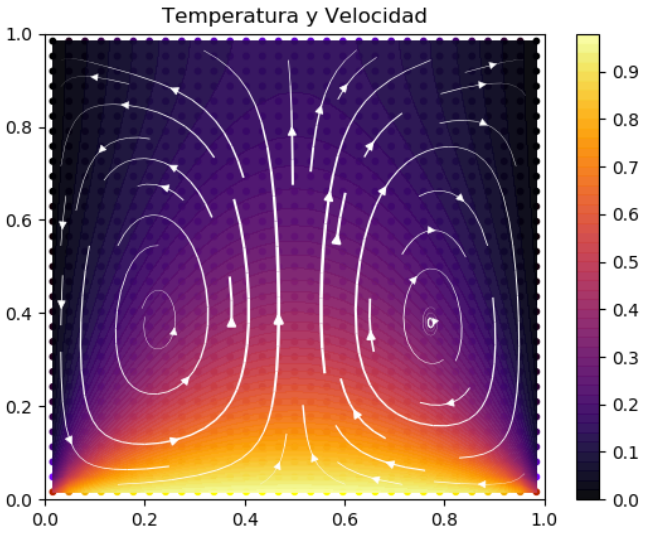
FIGURA 3.3: Resultados de la *Prueba 1* usando un $Ra = 10^3$. La figura A muestra el campo de velocidades y el color de fondo en la misma representa la temperatura de acuerdo con la barra de color anexa. La gráfica B muestra las curvas de nivel de la temperatura.

3.3.2. Segundo ejemplo de prueba

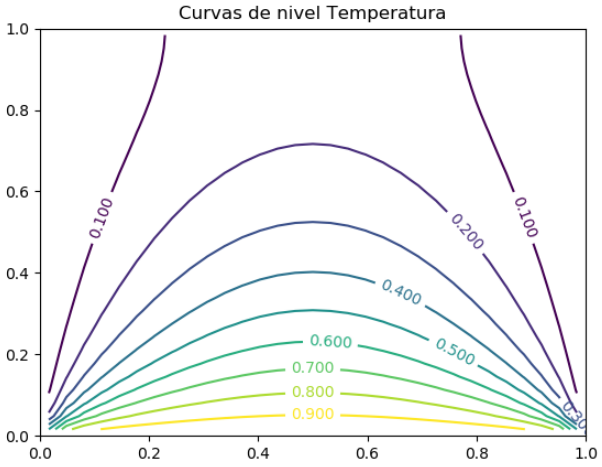
El segundo ejemplo es extraído de *Effects of thermal boundary conditions on natural convection flows within a square cavity* [12]. Este problema, se trata de un dominio cuadrado con la frontera superior adiabática y las fronteras laterales a una temperatura constante que es inferior a la temperatura (constante) de la base del cuadrado. Nos referiremos a este caso como *Prueba 2*, mismo que está definido por las siguientes condiciones de frontera:

$$\begin{aligned} U(0,Y) = U(1,Y) = U(X,0) = U(X,1) = 0; \\ V(0,Y) = V(1,Y) = V(X,0) = V(X,1) = 0; \\ \theta(X,0) = 1; \\ \theta(0,Y) = \theta(1,Y) = 0; \quad \frac{\partial \theta}{\partial Y}(X,1) = 0 \end{aligned} \tag{3.2}$$

Donde θ es la temperatura mientras que U y V son las componentes de la velocidad en la dirección de las coordenadas X y Y, respectivamente. Todas las variables anteriores están expresadas en su forma adimensional.
Los resultados de esta *Prueba 2* se muestran en la figura 3.4 y en esta se puede observar que de nuevo se obtienen resultados consistentes con los mostrados en la literatura [12], es decir, que las gráficas son visualmente (cualitativamente) muy similares. Esto tanto para el campo de velocidades (figura 3.4a), como para las curvas de nivel de la temperatura (figura 3.4b). El número de Nusselt promedio fue calculado sobre las fronteras izquierda ($X = 0$), derecha ($X = 1$) e inferior ($Y = 0$) obteniendose los valores de -2.66, 2.66 y 5.35, respectivamente. Que se tengan signos contrarios para los Nu promedio en las paredes laterales es debido a que la temperatura está fluyendo hacia afuera de la cavidad a través de estas. En este ejemplo, al igual que en el resto de simulaciones de la convección natural, se calculó la solución para un número de Rayleigh (Ra) de 10^3. Aunque en el trabajo con el modelo conceptual de esta prueba [12] no se reporta el valor exacto del Nu promedio, sí se muestran gráficas del Nu como función de Ra de las que se puede concluir una buena concordancia (discrepancia$< 5\,\%$). Solo se utilizó el valor de $Ra = 10^3$ porque corresponde al caso más manejable de los modelos (condiciones de frontera) establecidos como referencia. Además no es relevante para estas pruebas reproducir todos los pasos y resultados previos; por un lado, porque la métodología de las referencias y de la implementación no es exactamente la misma y, por el otro, porque el objetivo de dichas referencias en estas pruebas es solo el de **proveer modelos conceptuales coherentes** de convección natural. Así, se probaron y desarrollaton las funcionalidades del software necesarias para la convección natural (desacoplamiento SIMPLE, mallas recorridas, etcétera).



(A)



(B)

FIGURA 3.4: Resultados de la *Prueba 2* usando un $Ra = 10^3$. La figura A muestra el campo de velocidades y el color de fondo en la misma representa la temperatura de acuerdo con la barra de color anexa. La gráfica B muestra las curvas de nivel de la temperatura.

3.3.3. Tercer ejemplo de prueba

Por otro lado, como tercera validación, se tomó un ejemplo de *Numerical investigation of natural convection in a rectangular enclosure due to partial heating and cooling at vertical walls* [13], mismo que se denominará como *Prueba 3*. Las condiciones físicas de este ejemplo están definidas de acuerdo con la figura 3.5 y las siguientes condiciones de frontera:

$$\begin{aligned} U(0,Y) = U(1,Y) = U(X,0) = U(X,A) &= 0; \\ V(0,Y) = V(1,Y) = V(X,0) = V(X,A) &= 0; \\ \frac{\partial \theta}{\partial Y} &= 0; \quad \text{en Y=0 , Y=A} \\ \theta(0,Y) &= 1; \quad \text{en } 0 \leq Y \leq A/2 \\ \left.\frac{\partial \theta}{\partial X}\right|_{X=0} &= 0; \quad \text{en } A/2 \leq Y \leq A \\ \theta(1,Y) &= 0; \quad \text{en } A/2 \leq Y \leq A \\ \left.\frac{\partial \theta}{\partial X}\right|_{X=1} &= 0; \quad \text{en } 0 \leq Y \leq A/2 \end{aligned} \tag{3.3}$$

Los resultados de la *Prueba 3* se muestran en la figura 3.6, teniendo que la

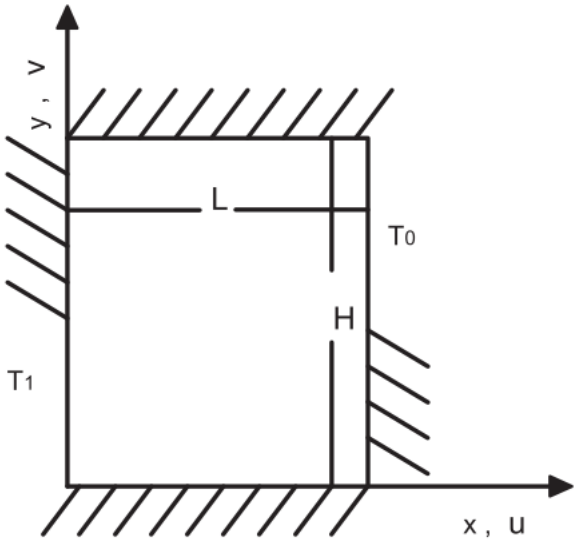
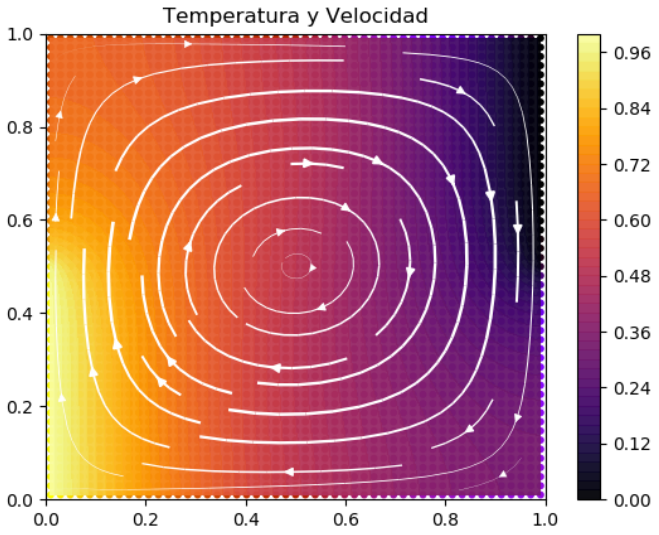
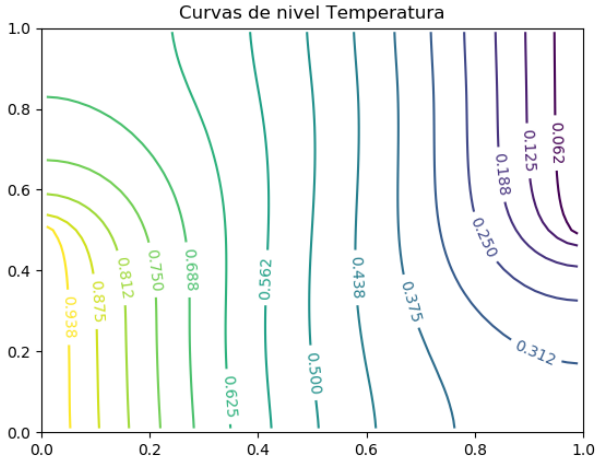


FIGURA 3.5: Esquema del dominio físico. Las secciones de la frontera marcadas con T_1 y T_0 indican temperatura constante mientras que el resto de la frontera es adiabática.

figura 3.6a representa el campo de velocidades y la figura 3.6b, las curvas de nivel de la temperatura. En este caso al calcular el *Nu* promedio sobre la mitad inferior de la frontera izquierda, la que se encuentra a temperatura T_1, se obtiene 1.39. La mitad superior de la frontera izquierda es adiabática, esto implica que el *Nu* promedio es 0 sobre esta región. Por lo tanto el promedio sobre toda la frontera izquierda es $Nu = 0.69$.



(A)



(B)

FIGURA 3.6: Resultados de la *Prueba 3* usando un $Ra = 10^3$. La figura A muestra el campo de velocidades y el color de fondo en la misma representa la temperatura de acuerdo con la barra de color anexa. La gráfica B muestra las curvas de nivel de la temperatura.

3.3.4. Cuarto ejemplo de prueba

También es muy relevante realizar la validación de un ejemplo en tres dimensiones, con ese objetivo la última validación realizada se tomó de *Natural convection in a cubic cavity: Implicit numerical solution of two benchmark problems*[14]. El problema a reproducir, al igual que en las pruebas anteriores, consta de las ecuaciones adimensionales de la convección natural utilizando la aproximación de Boussinesq; esta vez considerando las tres componentes espaciales. El problema se plantea sobre un dominio cúbico como el que se muestra en la figura 3.7 sometido a las siguientes condiciones de frontera, donde n indica la coordenada en la dirección normal a la superficie:

$$\theta = 1 \; en \; X = 0 \quad \theta = 0 \; en \; X = 1 \quad \frac{\partial \theta}{\partial n} = 0 \; en \text{ las otras 4 paredes.} \tag{3.4}$$

Suponiendo además, al igual que en las pruebas anteriores, condiciones de no deslizamiento; es decir, velocidades igual a cero sobre todas las paredes. Por tratarse de un dominio en tres dimenisiones, la visualización de resulta-

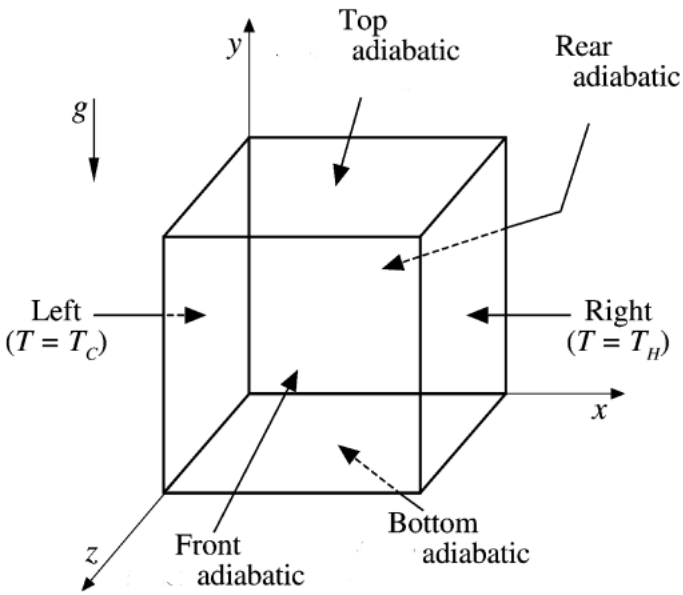
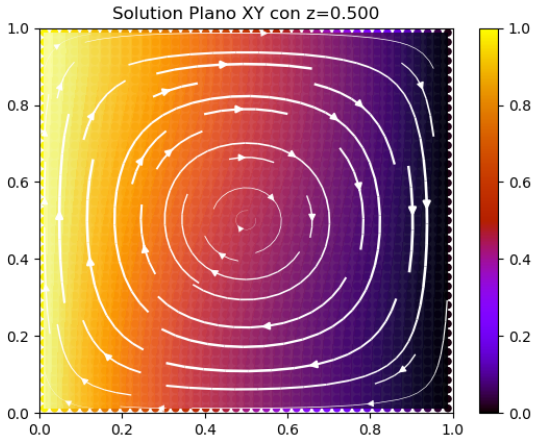


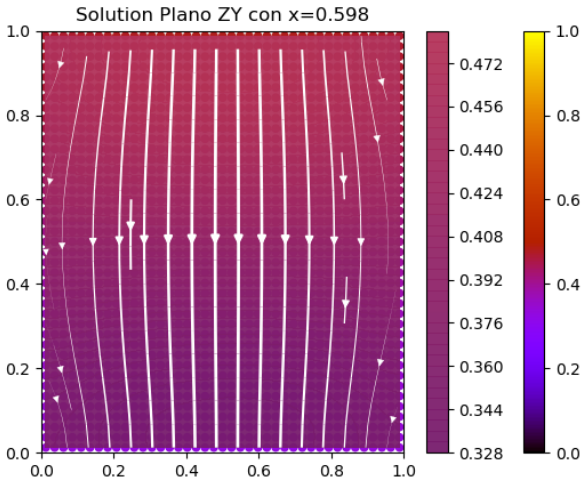
FIGURA 3.7: Esquema del dominio físico (cavidad cúbica) en la última prueba. Las secciones de la frontera marcadas con T_H y T_C indican temperatura constante mientras que el resto de la frontera es adiabática.

dos se realizó por medio de cortes. En este caso, para dar una clara visualización del resultado se muestran 6 cortes del dominio cúbico; tres cortes se muestran en la figura 3.8 y otros tres se presentan en la figura 3.9. Los cortes son en los planos z=0.5, x=0.598, y=0.524, z=0.598, x=0.402, y=0.207, para las figuras 3.8a, 3.8b, 3.8c, 3.9a, 3.9b, 3.9c; respectivamente. Con el conjunto de cortes se puede interpretar que la solución obtenida, en cuanto al campo de velocidades, es una serie de bucles en sentido horario que son paralelos al

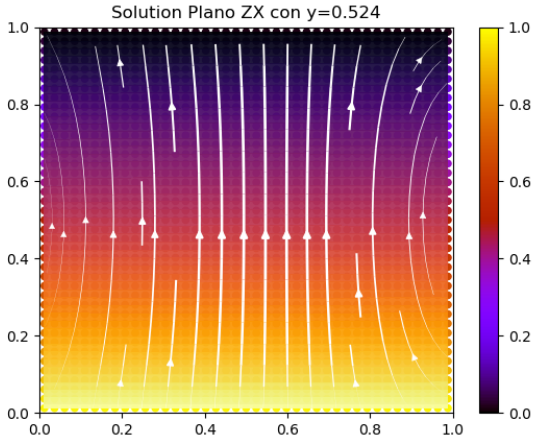
plano XY. Podemos decir que se tiene una situación similiar al primer ejemplo de prueba (sección 3.3.1) pero con una cavidad en tres dimensiones.



(A) $z = 0.5$

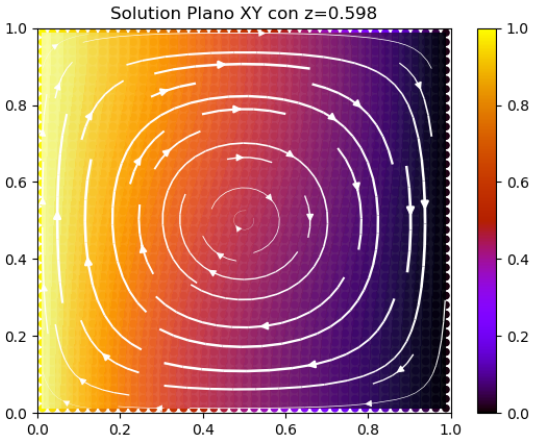


(B) $x = 0.598$

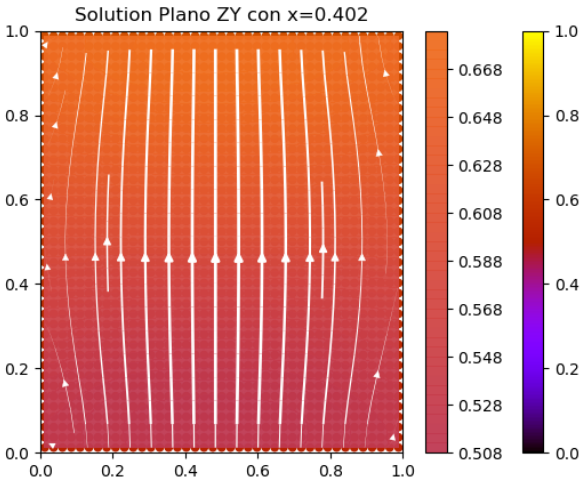


(C) $y = 0.524$

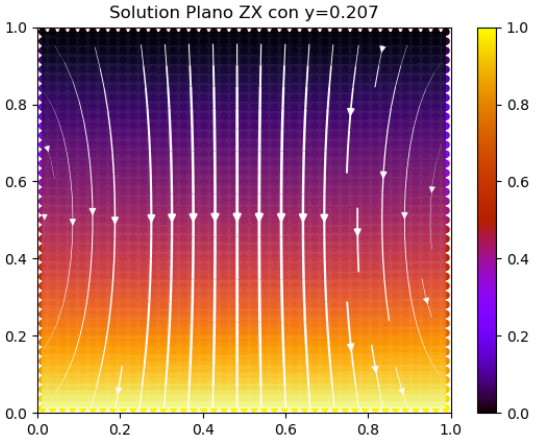
FIGURA 3.8: Resultados de la *Prueba 4* usando un $Ra = 10^3$. Las gráficas muestran el campo de velocidades con el color de fondo representando la temperatura. Las figuras A, B y C muestran un corte frontal, lateral y superior para $z = 0.5$, $x = 0.598$ y $y = 0.524$ respectivamente.



(A) $z = 0.598$



(B) $x = 0.402$



(C) $y = 0.207$

FIGURA 3.9: Resultados de la *Prueba 4* usando un $Ra = 10^3$. Las gráficas muestran el campo de velocidades con el color de fondo representando la temperatura. Las figuras A, B y C muestran un corte frontal, lateral y superior para $z = 0.598$, $x = 0.402$ y $y = 0.207$ respectivamente.

Capítulo 4

Modelación de la transferencia de calor en un invernadero

Las simulaciones mostradas en la sección anterior sirvieron para desarrollar las funcionalidades de software que son necesarias en los problemas de convección natural además de ser una primera validación los resultados que arroja el software. Sin embargo, en este trabajo se buscó también realizar simulaciones de modelos que puedan representar invernaderos. Esta sección muestra los resultados de los modelos propuestos para ese fin y resulta conveniente discutir brevemente la forma en la que se presentan dichos resultados.

4.1. Resultados y discución

Para empezar, las figuras que muestran las soluciones del campo de velocidades se muestran acompañadas de un esquema del modelo utilizado para generar la simulación. Por ejemplo, en la figura 4.1 se muestran juntos el esquema con las medidas del modelo (figura 4.1a) y la respectiva gráfica con el campo de de velocidades obtenido del mismo (figura 4.1b). También es relevante recordar que los modelos propuestos constan de un rectángulo de fronteras a temperatura constante Ta que tiene inmerso paredes que definen un invernadero con un techo y suelo a temperaturas constantes Tr y Ts junto con paredes laterales adiabáticas ($f_{wall} = 0$) y una abertura que representa una ventilación natural. Esto es suficiente para interpretar los esquemas presentados aunque más detalles del modelo conceptual se pueden revisar en la sección 2.1 . Por lo que se refiere a las gráficas que muestran el campo de velocidades, como la figura 4.1b, cabe mencionar que el color de fondo corresponde a la distribución de temperatura de acuerdo a la barra de colores que se muestra en dichas gráficas. Para visualizar mejor la solución de la variable temperatura se muestran también figuras, como la figura 4.3, que exponen la distribución de temperatura por medio de gráficas de contorno (curvas de nivel) y que sirven además para contrastar la solución de temperatura dependiendo de la posición de la ventilación en el invernadero. Los modelos empleados solo varían en su geometría considerando 3 tipos de techo; plano,circular o triangular y 2 tipos de ventilación; ventilación en techo o ventilación en pared lateral. En las tres secciones que siguen se presentan

los resultados y su discusión por cada uno de los tres tipos de forma de techo. En todas las simulaciones se usaron variables adimensionales así como números de Rayleigh y de Prandtl con valores de 10^3 y 0.71, respectivamente.

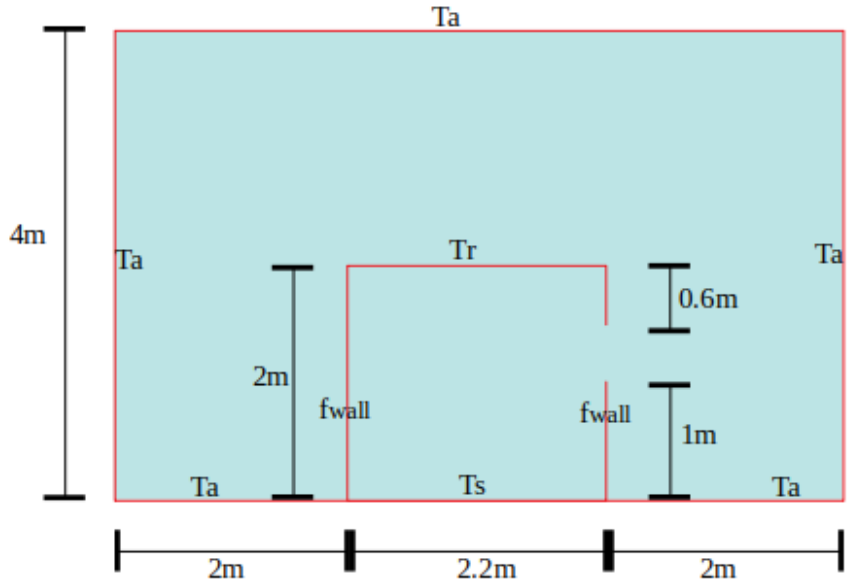
4.1.1. Resultados de invernadero con techo plano

Los primeros resultados que se exponen son los del invernadero con techo plano y ventilación lateral. En 4.1a se puede ver la escala real del modelo pero para los resultados se usa la escala adimensional. En efecto, el invernadero en el modelo a escala real tiene una altura de 2 metros y ancho de 2.2 metros, mientras que en los resultados de la figura 4.1b presenta una altura y ancho de 0.5 y 0.55, respectivamente. Esto representa una de las ventajas de las variables adimensionales pues así ya no es tan relevante las medidas exactas del modelo sino las proporciones entre dichas medidas. Del campo de velocidades (figura 4.1b) podemos decir que dentro del invernadero se forma un bucle similar a los obtenidos para las cavidades en la sección *Pruebas de validez de resultados a través de modelos conocidos*(sección 3.3). Fuera del invernadero se presenta un fenómeno que es primordialmente difusivo. El invernadero de techo plano pero con ventilación en techo presenta resultados similares (ver figura 4.2); sin embargo, la distribución de la temperatura es distinta, como lo expone la figura 4.3.
Para tener un cálculo más cuantitativo de dichas diferencias, se calcularon los números de Nusslet promedio en tres regiones, más específicamente, tres segmentos de recta. A saber, una recta que abarca el suelo del invernadero, una recta vertical que va del centro del suelo hasta el centro del techo y una recta horizontal que abarca el invernadero a una altura constante de $Y = 0.25$. Para futuras referencias, estas tres líneas serán denominadas *suelo*,*vertical mid* y *horizontal mid*; respectivamente. De esta manera, los *Nu* promedio calculados para ambos tipos de ventilación se muestran en la tabla 4.1. Con los valores obtenidos del *Nu(vertical mid)* se puede decir que al pasar de

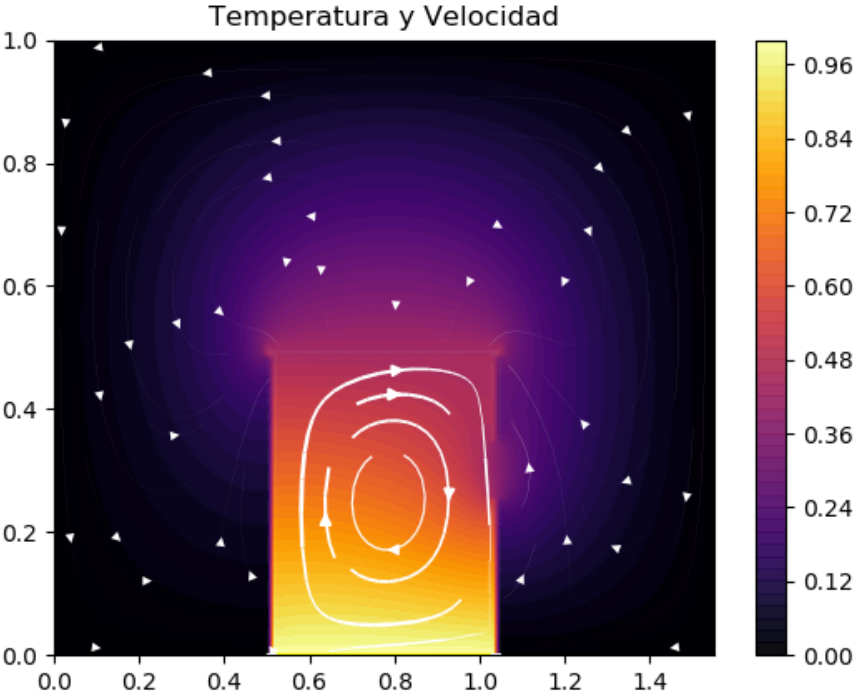
Ventilación	*Nu*(suelo)	*Nu*(horizontal mid)	*Nu*(vertical mid)
Lateral	1.387	1.336	0.051
En techo	1.182	1.217	-0.067

TABLA 4.1: Los números de Nusselt promedio obtenidos en distintas rectas para los dos tipos de ventilación en el modelo de invernadero con techo plano.

una ventilación a otra el flujo en el centro pasa de ir ligeramente hacia la derecha (+) a ir ligeramente a la iquierda (-). Estos valores son pequeños porque el flujo de temperatura es primordialmente vertical y, en ese sentido, el flujo vertical en el suelo (Nu_{suelo}) varía cerca de un 17 % debido al cambio en la ventilación.

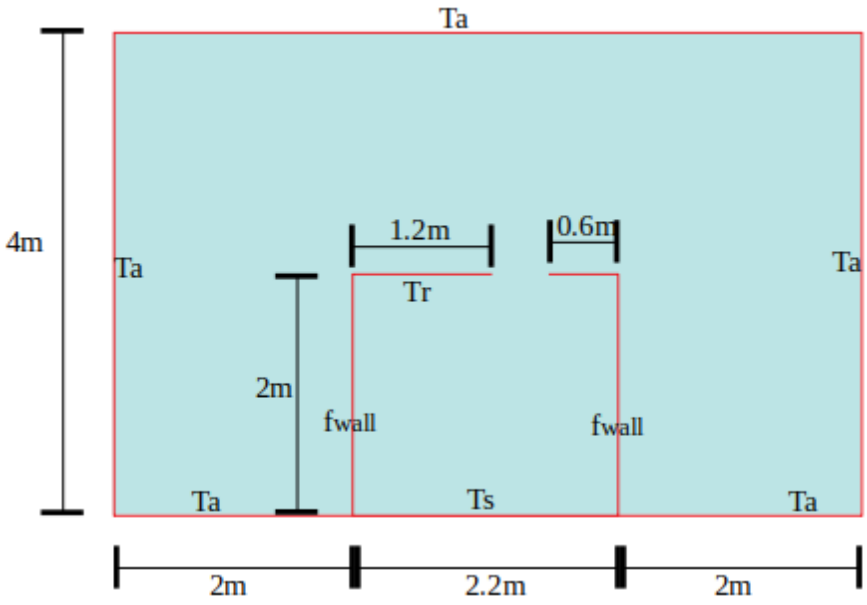


(A) Modelo

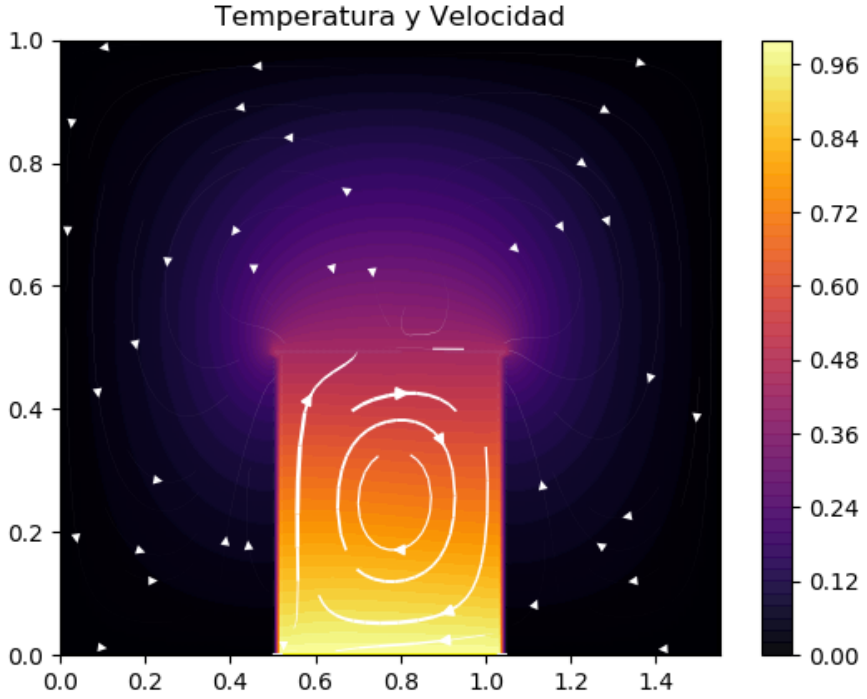


(B) Velocidad y temperatura.

FIGURA 4.1: Resultados del modelo 1 de invernadero: ventilación lateral y techo plano. En la figura (A) se muestra el modelo y en la figura (B) el campo de velocidades con el color de fondo representando la temperatura de acuerdo con la escala lateral.

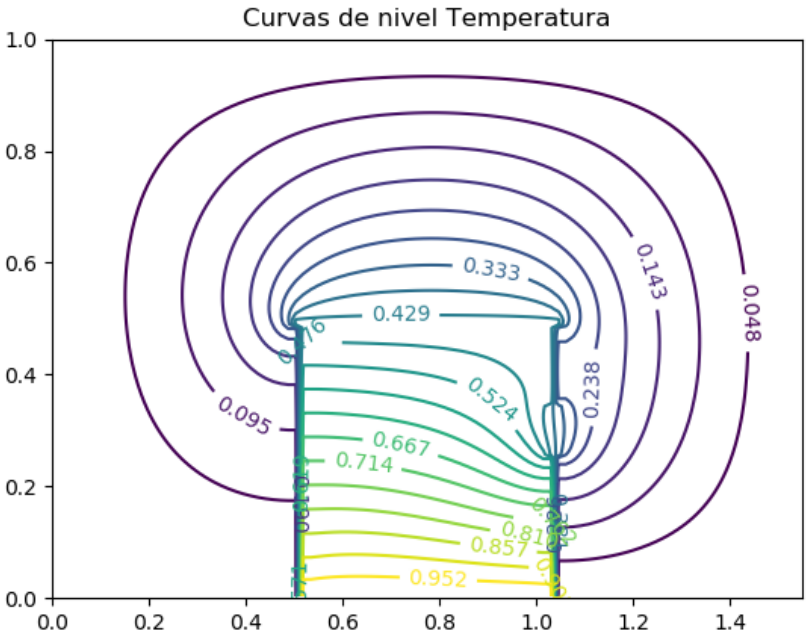


(A) Modelo

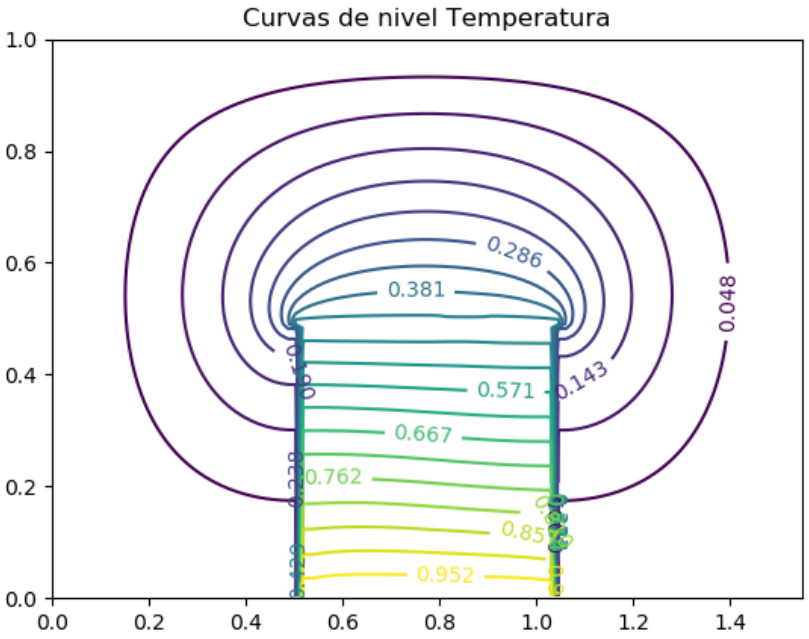


(B) Velocidad y temperatura.

FIGURA 4.2: Resultados del modelo 2 de invernadero: ventilación en alto y techo plano. En la figura (A) se muestra el modelo y en la figura (B) el campo de velocidades con el color de fondo representando la temperatura de acuerdo con la escala lateral.



(A)



(B)

FIGURA 4.3: Gráficas de curvas de nivel para los invernaderos de techo plano con (A) ventilación lateral y (B) ventilación en alto.

4.1.2. Resultados de invernadero con techos circular y triangular

Los resultados del invernadero de techo circular con ventilación lateral y superior se pueden observar en las figuras 4.4 y 4.5, respectivamente. La comparación entre ambos casos se da con las gráficas de curvas de nivel que se exhiben en la figura 4.6 y con los resultados de la tabla 4.2.

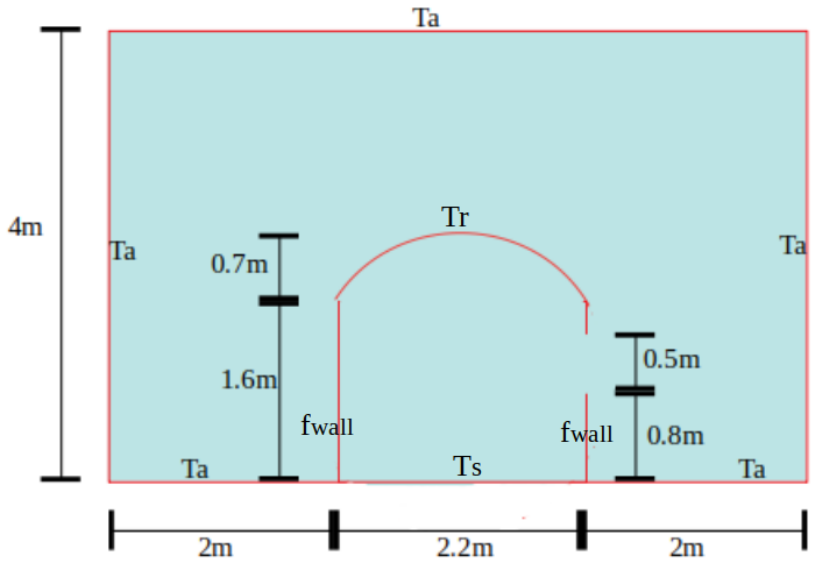
De forma análoga los resultados del modelo con techo en forma triangular se muestran en las figuras 4.8, 4.7, 4.9 y la tabla 4.3.
A grandes rasgos podemos decir que en estas geometrías de techo (circular y triangular) los resultados son similares a los resultados del invernadero con techo plano. De los tres tipos de techos, el triangular es el que fué más propenso a divergir, esto en el sentido de que las magnitudes del campo de velocidades crecen de manera descontrolada. Así, en las primeras iteraciones la condición de continuidad (ver sección 2.3.3) disminuye acercandose a cero, pero rápidamente, cerca de la iteración número 60, la condición de continuidad aumenta alejandose del valor buscado. En las conclusiones de [25] se afirma que los confinamientos de paredes cuadradas son mucho más fáciles de modelar y sus patrones térmicos e hidrodinámicos son menos complejos en comparación con paredes inclinadas. Por otro lado, intuitivamente se puede esperar que una geometría "*en punta*"sea más difícil de tratar dado que en esa región el campo de velocidades debe cambiar de forma abruta para seguir la pared en punta.

Ventilación	*Nu*(suelo)	*Nu*(horizontal mid)	*Nu*(vertical mid)
Lateral	1.475	1.217	0.083
En techo	1.201	1.234	-0.031

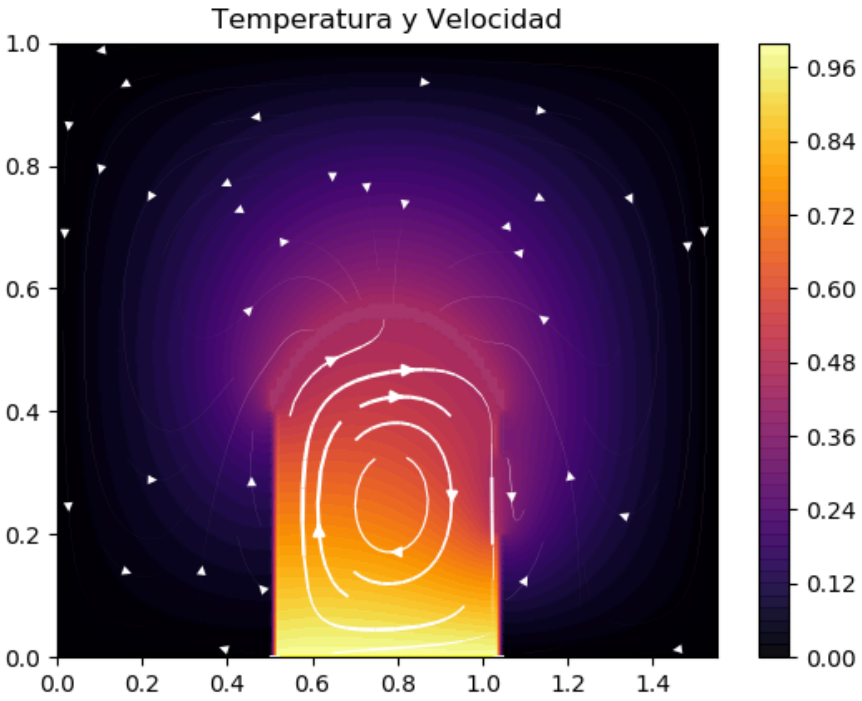
TABLA 4.2: Los números de Nusselt promedio obtenidos en distintas rectas para los dos tipos de ventilación en el modelo de invernadero con techo circular.

Ventilación	*Nu*(suelo)	*Nu*(horizontal mid)	*Nu*(vertical mid)
Lateral	1.823	1.346	0.030
En techo	1.541	1.536	-0.089

TABLA 4.3: Los números de Nusselt promedio obtenidos en distintas rectas para los dos tipos de ventilación en el modelo de invernadero con techo triangular.

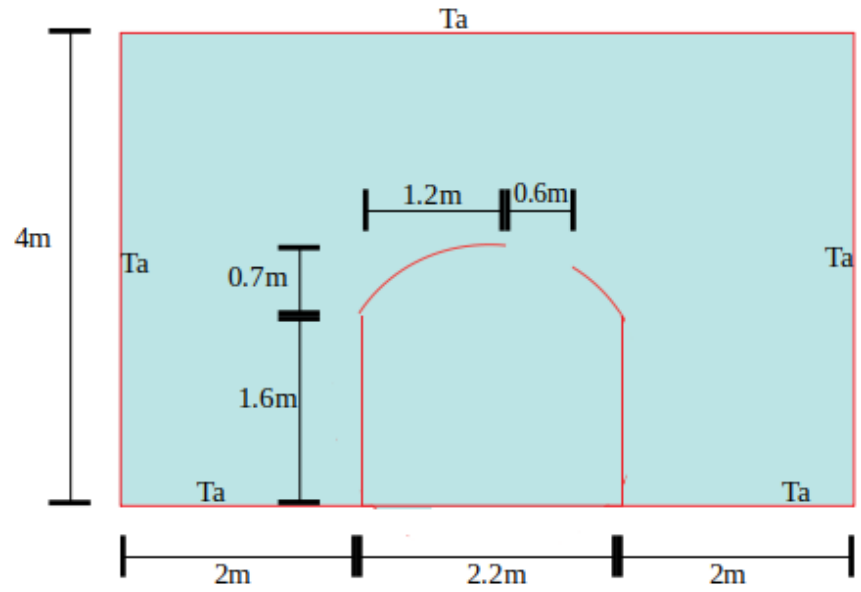


(A) Modelo

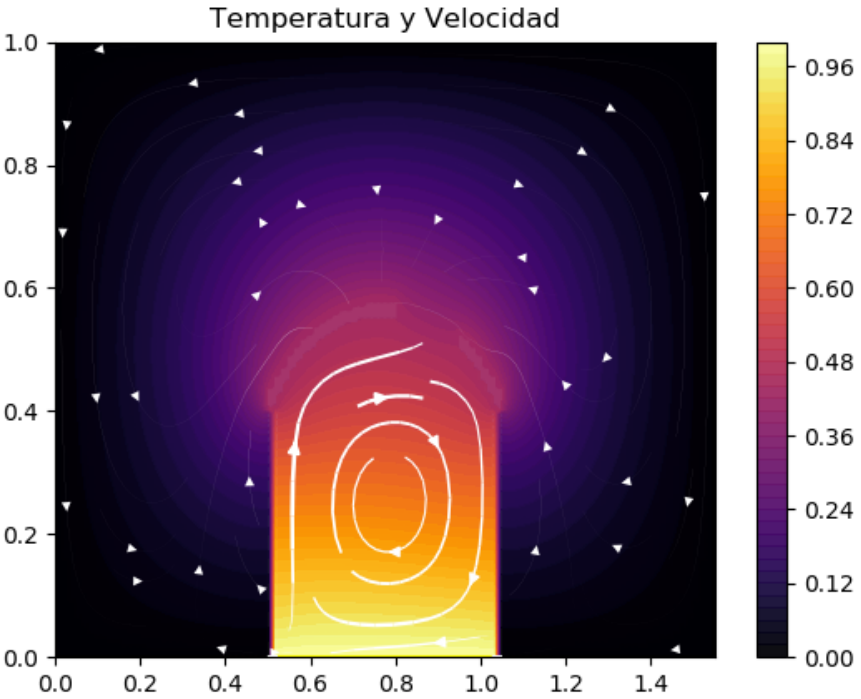


(B) Velocidad y temperatura.

FIGURA 4.4: Resultados del modelo 1 de invernadero: ventilación lateral y techo circular. En la figura (A) se muestra el modelo y en la figura (B) el campo de velocidades con el color de fondo representando la temperatura de acuerdo con la escala lateral.

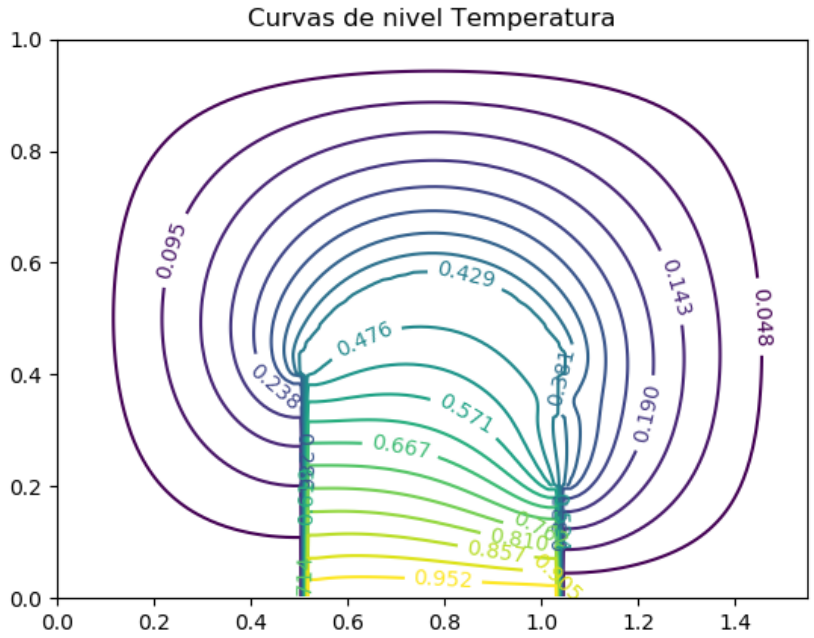


(A) Modelo

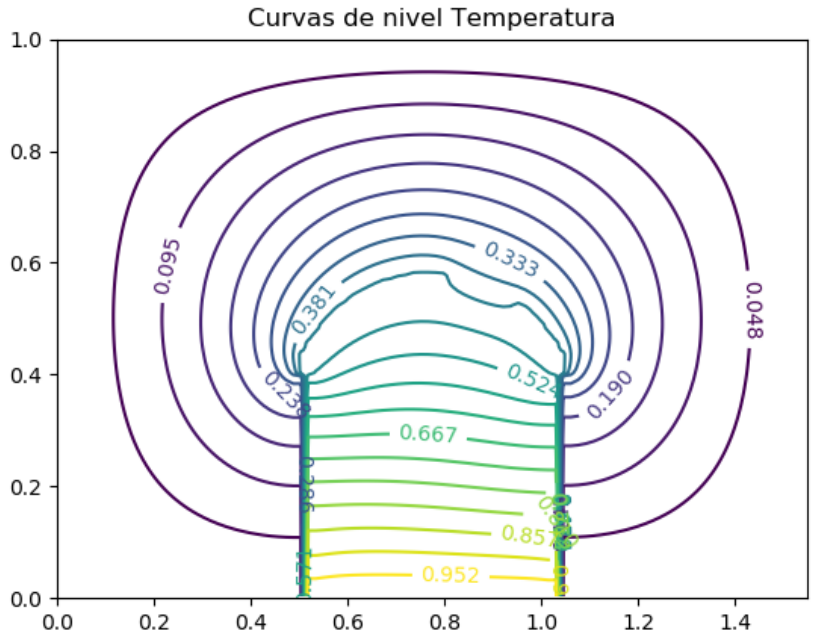


(B) Velocidad y temperatura.

FIGURA 4.5: Resultados del modelo 2 de invernadero: ventilación en alto y techo circular. En la figura (A) se muestra el modelo y en la figura (B) el campo de velocidades con el color de fondo representando la temperatura de acuerdo con la escala lateral.

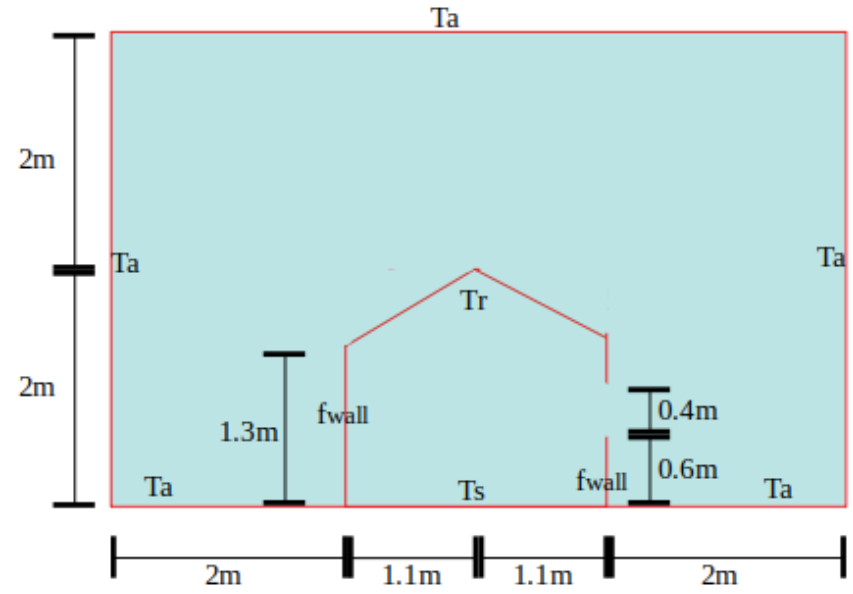


(A)

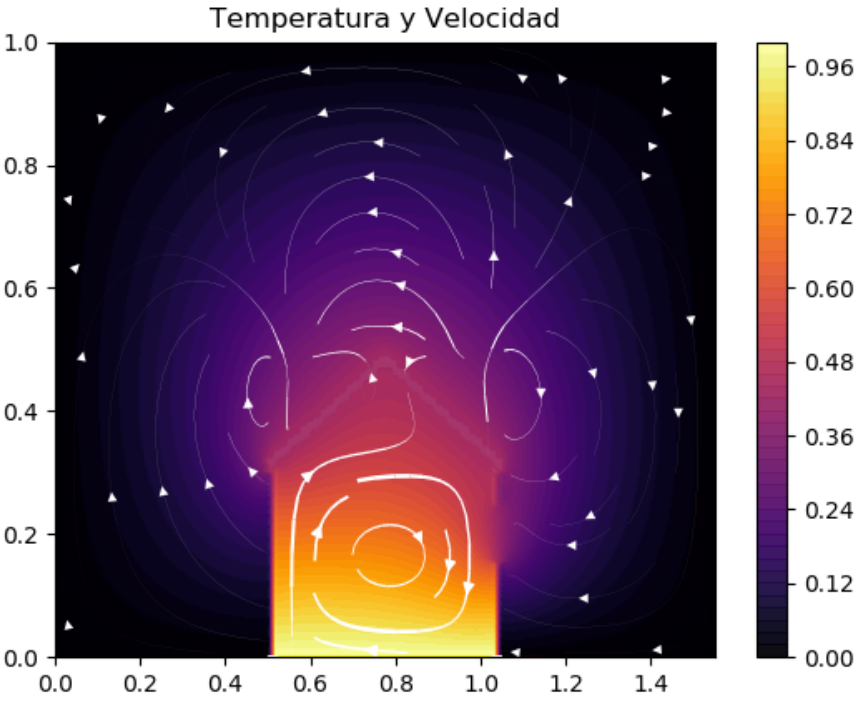


(B)

FIGURA 4.6: Gráficas de curvas de nivel para los invernaderos de techo circular con (A) ventilación lateral y (B) ventilación en alto.

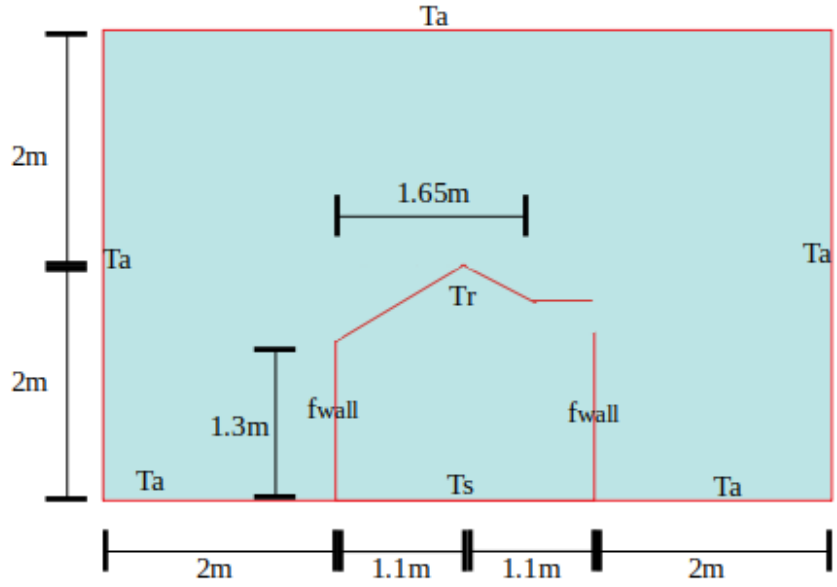


(A) Modelo

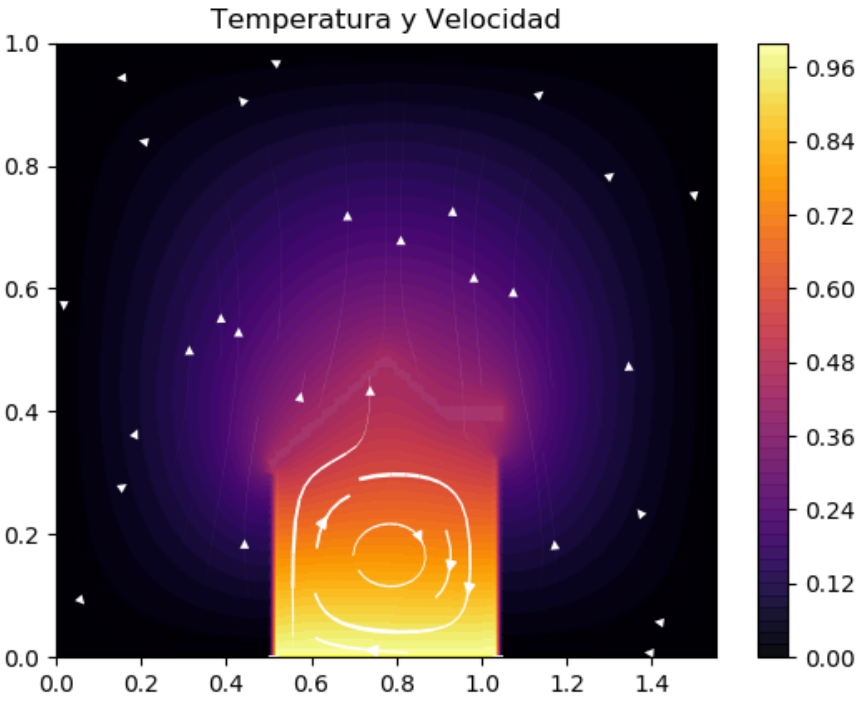


(B) Velocidad y temperatura.

FIGURA 4.7: Resultados del modelo 2 de invernadero: ventilación en alto y techo triangular. En la figura (A) se muestra el modelo y en la figura (B) el campo de velocidades con el color de fondo representando la temperatura de acuerdo con la escala lateral.

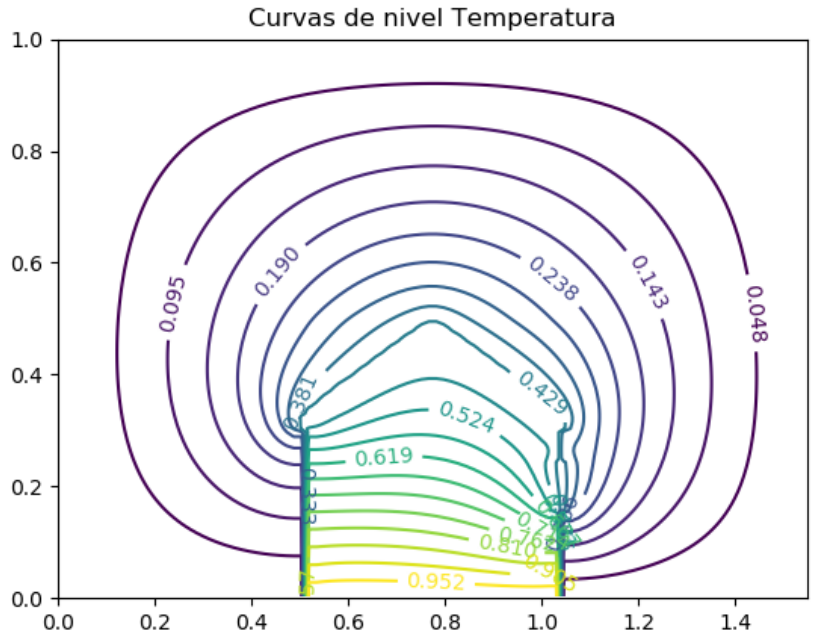


(A) Modelo

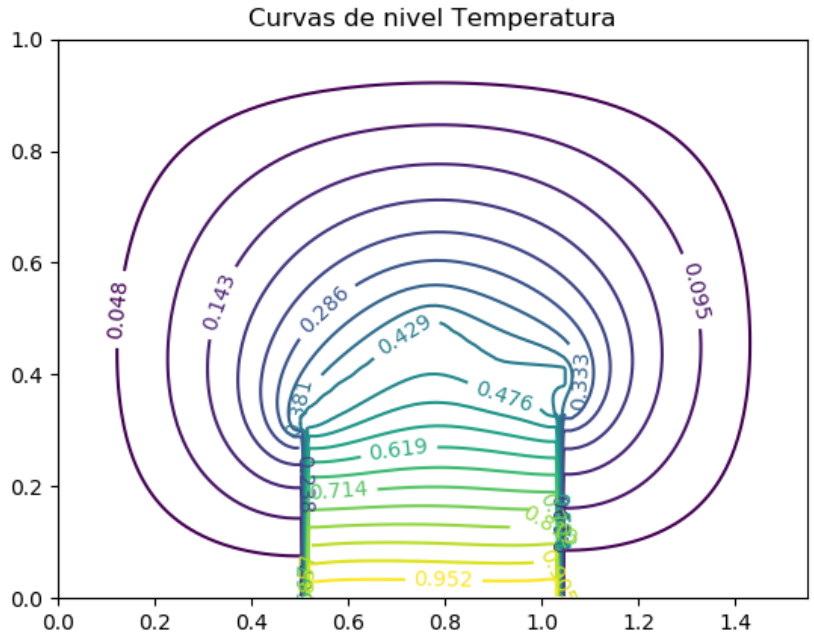


(B) Velocidad y temperatura.

FIGURA 4.8: Resultados del modelo 5 de invernadero: ventilación lateral y techo triangular. En la figura (A) se muestra el modelo y en la figura (B) el campo de velocidades con el color de fondo representando la temperatura de acuerdo con la escala lateral.



(A)



(B)

FIGURA 4.9: Gráficas de curvas de nivel para los invernaderos de techo triangular con (A) ventilación lateral y (B) ventilación en alto.

4.2. Inspección inicial de tiempos de ejecución

Con el objetivo de identificar las áreas de mayor demanda computacional se realizaron pruebas enfocadas al tiempo de ejecución de distintas secciones del código. Se tomó como base el ejemplo del invernadero con techo circular y ventilación lateral, el cual utililza una malla de 15000 volúmenes de control y posee un tiempo total de ejecución de 327.44s. De este tiempo un pequeño porcentaje se emplea en definir las mallas con sus condiciones de frontera pero el 99 % del tiempo de ejecución, se usa para desacoplar las variables de presión, temperatura y componentes de la velocidad (P,T,U,V) usando el método SIMPLE. Lo que se busca evidenciar con este hecho es que para optimizar el código o, al menos reducir su tiempo de ejecución es necesario enfocarse en la sección de código del método SIMPLE. Esta sección corresponde a un ciclo for en el que se realizan cálculos para obtener aproximaciones sucesivas de la presión, temperatura y componentes de la velocidad (P,T,U,V). Así, de los 316s que se invierten en el ciclo for del SIMPLE, 78.62s se utilizan para hacer los cálculos (preparación y solución de sistema de ecuaciones) que resultan en el valor final de la temperatura T. Este tiempo comprende el 24.10 % de todo el tiempo que abarca dicho ciclo y un tiempo promedio por cada iteración de 0.6497 ± 0.2346s. En la tabla 4.4 se muestra como se distribuye el tiempo de cálculo de cada variable y de aquí podemos observar que la ecuación que más tiempo consume es la ecuación de corrección a la presión. El porcentaje que no se muestra en dicha tabla corresponde al tiempo empleado en verificar la condición de continuidad y el error que muestra el $t_{promedio}$ corresponde al doble de la desviación estándar.
Los cálculos más importantes que se realizan para cada variable, en cada ite-

Variable	t_{total}	% SIMPLE	$t_{promedio}$
U	35.89s	11.00 %	0.2967 ± 0.0741 s
V	35.51s	10.88 %	0.2934 ± 0.0761 s
P	176.13s	53.99 %	1.456 ± 0.4545 s
T	78.62s	24.10 %	0.6497 ± 0.2346 s

TABLA 4.4: Tabla que muestra, por cada una de las variables (U,V,P y T) que se resuelven, el tiempo total invertido en el ciclo for (T_{total}), el porcentaje (% SIMPLE) que ese tiempo representa con respecto al tiempo de todo el bucle y el tiempo promedio de cálculo en cada iteración ($T_{promedio}$); usando GMRES (librería scipy).

ración, consisten básicamente en: *crear un nuevo sistema de ecuaciones, calcular un precondicionador para la matriz generada y resolver el sistema de ecuaciones,* en ese orden. De aquí, se exploraron algunas formas en las que se podría impactar en el tiempo de ejecución del código.
En primer lugar, la solución del sistema de ecuaciones corresponde a la parte que más tiempo demanda. En el caso de la variable P, el 80 % del tiempo invertido en dicha variable se ocupa en aplicar GMRES. Al hacer que en todas

las variables (U,V,P,T) se cambie de GMRES a BICGSTAB el tiempo se reduce hasta 232.94s (71.4 % del tiempo original). Es relevante hacer notar que con

Variable	t_{total}	% SIMPLE	$t_{promedio}$
U	34.56s	14.84 %	0.2856 ± 0.0428 s
V	33.77s	14.50 %	0.2791 ± 0.0153 s
P	92.13s	39.55 %	0.7614 ± 0.1279 s
T	72.43s	31.10 %	0.5986 ± 0.100 s

TABLA 4.5: Tabla que muestra, por cada una de las variables (U,V,P y T) que se resuelven, el tiempo total invertido en el ciclo for (T_{total}), el porcentaje (% SIMPLE) que ese tiempo representa con respecto al tiempo de todo el bucle y el tiempo promedio de cálculo en cada iteración ($T_{promedio}$); usando BICGSTAB (librería scipy).

este cambio, el único tiempo que cambia sustancialmente es el de la variable P (ver tabla 4.5). Este hecho puede deberse a que esta es la variable que se está utilizando para actualizar a las demás y un algoritmo de solución de sistemas de ecuaciones puede adaptarse mejor a este hecho en comparación a otro. Como hemos dicho, con el uso del algoritmo BICSTAB para resolver los sistemas de ecuaciones, la disminución del tiempo total es notoria. En efecto, la parte que más tiempo consume es la solución de sistemas de ecuaciones lineales y con esta motivación, se decidió implementar una versión de BICGSTAB. Para poder competir, en cuanto a tiempo, con la implementación de scipy, la implementación de BICGSTAB que fue desarrollada se basó en la librería NUMBA. Esta librería provee una compilación *just in time* y además se utilizaron herramientas básicas de paralelismo de esta misma librería. En este aspecto, se observó que para que la librería NUMBA impactara en el tiempo de ejecución fué necesario utilizar dicha herramienta para funciones cortas y sobre todo con operaciones entre objetos simples de python o arreglos de Numpy. Por otro lado, se observó que los sistemas de las variables U y V pueden alcanzar la convergencia sin necesidad de un precondicionador e inclusive, que se pueden utilizar las soluciones de la iteración anterior como vector inicial del GMRES o BICSTAB para disminuir el tiempo en el que se llega a la solución debido a que se parte de un vector cercano a la misma.
Al final, al usar la implementación propia de BICGSTAB que toma como aproximación inicial la solución anterior y sólo precondiciona las variables P y T se obtiene un tiempo de 171.40s lo que representa 52.6 % del tiempo original. Dado que este tiempo relativamente pequeño, (aproximadamente 3 minutos) se diría que, al menos para la simulación realizada, Python puede proveer un tiempo de ejecución suficiente. Si en vez de usar una malla con 15000 volúmenes de control se utiliza una malla de 124700 volúmenes el tiempo es de 46 minutos. Esto, para los alcances del presente trabajo, sigue siendo un tiempo aceptable.

Capítulo 5

Conclusiones

En resumen, se ha desarrollado un software basado en un paradigma de programación orientado a objetos que es capaz de implementar el FVM de una forma accesible, reutilizable y con una construcción basada en la abstracción del método. El software es reutilizable en el sentido que puede usarse para resolver una variedad de planteamientos físicos y porque la utilización de *Python* lo vuelve cómodo para una comunidad más grande en comparación con otros lenguajes de programación.
Existen otros softwares que también son orientados a la dinámica de fluidos computacional (CFD). Dentro de los softwares comerciales podemos encontrar ANSyS, FLUENT o COMSOL; mientras que, por el lado de los libres están algunos como OpenFOAM, LibMesh o Dune. A diferencia de estos sistemas, la implementación desarrollada es un software que está hecho completamente en Python y esto permite explorar las bondades del lenguaje en términos de eficiencia.
En efecto, los resultados obtenidos pueden servir como indicador de la viabilidad, en cuanto a costo/beneficio, de utilizar Python para cómputo numérico mediante herramientas de compilado *just in time* y paralelismo, como lo es Numba. Además, la descripción de la implementación puede resultar relevante pues en la mayoría de los textos consultados la discusión de los aspectos computacionales de la simulación era mínima.
Por otro lado, las reproducciones de problemas de convección natural en una cavidad muestran una buena correspondencia con los resultados teóricos. Esta afirmación se debe, a que los números de Nusselt así como las distribuciones de temperatura y campos de velocidad son congruentes con las referencias citadas. Ambos aspectos, tanto el cualitativo (gráficas de distribucion de temperatura y campos de velocidad), como el cualitativo (número de Nusselt), son necesarios para permitir establecer confianza en los resultados.
El punto débil aquí es que sólo se obtuvieron simulaciones coherentes para un $Ra = 10^3$, sin embargo, el objetivo principal de estos ensayos no fue el de reproducir todos los pasos y resultados previos sino el de validar los métodos programados específicamente para problemas de convección natural como el desacoplamiento de ecuaciones o las mallas recorridas. Si bien es cierto que, para tener una buena validación es preferible comparar con soluciones analíticas o valores experimentales, en este caso no se contaba con ellos. Por lo que se compararon los resultados con otras referencias para sustentar la idea de que el modelo computacional está correctamente implementado. Aún así,

para que futuras iteraciones del desarrollo sean capaces de reproducir situaciones físicas más cercanas a la situación real, es clave revisar el modelo numérico en busca de aproximaciones de mayor orden de precisión.
En la modelación de la transferencia de calor en invernaderos se observan resultados coherentes con la intuición. Además, los números de *Nusselt* promedio calculados ayudan a evidenciar que existen diferencias en la distribución de temperatura como consecuencia de cambiar la posición de la ventilación natural. En particular, resulta relevante que se observen diferencias claras en el *Nu* promedio calculado en el suelo pues, este es el lugar donde se localizarían los cultivos.
En el presente trabajo se busca que los modelos simbolicen la distribución de temperatura de invernaderos debido a dos motivos; por un lado, existen ya textos académicos en este sentido que prueban el interés en este tipo de simulaciones y, por el otro, el software desarrollado tendría el potencial para aplicaciones. En efecto, los invernaderos representan una situación en la que es importante mantener el control de la temperatura, ventilación y otras variables climáticas. En este sentido, se debe continuar desarrollando el software, sobre todo en la validación de resultados con datos experimentales, y así poder tener aplicaciones en el diseño de invernaderos o en el monitoreo de las variables físicas dentro del mismo.
Por último, además de lo ya mencionado, una posible modificación al software sería la incorporación de una clase o módulo que funcionara como lector de archivos generados en un software externo (como gmsh) donde fuera posible generar la malla y su etiquetado.

Bibliografía

[1] *Objetivos de desarrollo sostenible de Naciones Unidas*, https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/, Accesado: 28-nov-2018.

[2] *Objetivos de desarrollo sostenible:* Hambre cero, https://www.un.org/sustainabledevelopment/es/hunger/, Accesado: 28-nov-2018.

[3] *The greenhouse advantage*, https://www.growingformarket.com/articles/The-greenhouse-advantage, Accesado: 26-nov-2018.

[4] E. Santolini, B. Pulvirenti, S. Benni, L. Barbaresi, D. Torreggiani y P. Tassinari, «Numerical study of wind-driven natural ventilation in a greenhouse with screens», *Computers and Electronics in Agriculture*, vol. 149, n.º September 2017, págs. 41-53, 2018, ISSN: 01681699. DOI: 10.1016/j.compag.2017.09.027. arXiv: 1606.00586. dirección: https://doi.org/10.1016/j.compag.2017.09.027.

[5] P. E. Bournet y T. Boulard, «Effect of ventilator configuration on the distributed climate of greenhouses: A review of experimental and CFD studies», *Computers and Electronics in Agriculture*, vol. 74, n.º 2, págs. 195-217, 2010, ISSN: 01681699. DOI: 10.1016/j.compag.2010.08.007. dirección: http://dx.doi.org/10.1016/j.compag.2010.08.007.

[6] D. K. Fidaros, C. A. Baxevanou, T. Bartzanas y C. Kittas, «Numerical simulation of thermal behavior of a ventilated arc greenhouse during a solar day», *Renewable Energy*, vol. 35, n.º 7, págs. 1380-1386, 2010, ISSN: 09601481. DOI: 10.1016/j.renene.2009.11.013. dirección: http://dx.doi.org/10.1016/j.renene.2009.11.013.

[7] F. D. Molina-Aiz, H. Fatnassi, T. Boulard, J. C. Roy y D. L. Valera, «Comparison of finite element and finite volume methods for simulation of natural ventilation in greenhouses», *Computers and Electronics in Agriculture*, vol. 72, n.º 2, págs. 69-86, 2010, ISSN: 01681699. DOI: 10.1016/j.compag.2010.03.002. dirección: http://dx.doi.org/10.1016/j.compag.2010.03.002.

[8] D. Piscia, P. Muñoz, C. Panadès y J. I. Montero, «A method of coupling CFD and energy balance simulations to study humidity control in unheated greenhouses», *Computers and Electronics in Agriculture*, vol. 115, págs. 129-141, 2015, ISSN: 01681699. DOI: 10.1016/j.compag.2015.05.005. dirección: http://dx.doi.org/10.1016/j.compag.2015.05.005.

[9] D. Piscia, J. I. Montero, E. Baeza y B. J. Bailey, «A CFD greenhouse night-time condensation model», *Biosystems Engineering*, vol. 111, n.º 2, págs. 141-154, 2012, ISSN: 15375110. DOI: 10.1016/j.biosystemseng.2011.11.006. dirección: http://dx.doi.org/10.1016/j.biosystemseng.2011.11.006.

[10] G. Tong, D. M. Christopher y G. Zhang, «New insights on span selection for Chinese solar greenhouses using CFD analyses», *Computers and Electronics in Agriculture*, vol. 149, págs. 3-15, 2018, ISSN: 01681699. DOI: 10.1016/j.compag.2017.09.031. arXiv: 1606.00586. dirección: https://linkinghub.elsevier.com/retrieve/pii/S0168169916308481.

[11] G. D. V. Davis, «Natural convection of air in a square cavity: A bench mark numerical solution», *International Journal for Numerical Methods in Fluids*, vol. 3, n.º 3, págs. 249-264, jun. de 1983, ISSN: 1097-0363. DOI: 10.1002/fld.1650030305. dirección: https://doi.org/10.1002/fld.1650030305.

[12] T. Basak, S. Roy y A. R. Balakrishnan, «Effects of thermal boundary conditions on natural convection flows within a square cavity», *International Journal of Heat and Mass Transfer*, vol. 49, n.º 23-24, págs. 4525-4535, 2006, ISSN: 00179310. DOI: 10.1016/j.ijheatmasstransfer.2006.05.015.

[13] P. Alam, A. Kumar, S. Kapoor y S. R. Ansari, «Numerical investigation of natural convection in a rectangular enclosure due to partial heating and cooling at vertical walls», *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, n.º 6, págs. 2403-2414, 2012, ISSN: 10075704. DOI: 10.1016/j.cnsns.2011.09.004. dirección: http://dx.doi.org/10.1016/j.cnsns.2011.09.004.

[14] C. Taylor y F. Group, «NATURAL CONVECTION IN A CUBIC CAVITY : IMPLICIT NUMERICAL SOLUTION OF TWO BENCHMARK PROBLEMS», n.º January, págs. 99-123, 2006. DOI: 10.1080/10407780600605195.

[15] L. M. de la Cruz Salas, «Computo paralelo en la solucion numerica de las ecuaciones de balance en flujo turbulento», Tesis doct., Universidad Nacional Autónoma de México, 2005.

[16] H. K. Versteeg y W Malalasekera, *An Introduction to Computational Fluid Dynamics*. 2007, vol. M, pág. 517, ISBN: 9780131274983. dirección: http://www.pearson.ch/HigherEducation/PrenticeHall/1471/9780131274983/An-Introduction-to-Computational.aspx.

[17] Y. Saad, «Iterative methods for sparse linear systems», *IEEE Computational Science and Engineering*, vol. 3, n.º 4, págs. xviii + 528, 2003, ISSN: 1070-9924.

[18] H. Van der Vorst, «Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems», *SIAM Journal on Scientific and Statistical Computing*, vol. 13, pág. 631, mar. de 1992. DOI: 10.1137/0913035.

[19] J. R. Shewchuk, «An Introduction to the Conjugate Gradient Method Without the Agonizing Pain», págs. 133-150, 1994. DOI: 10.2307/j.ctvcm4h3p.12.

[20] M. H. Schultz e Y Saad, «GMRES: A GENERALIZED MINIMAL RESIDUAL ALGORITHM FOR SOLVING NONSYMMETRIC LINEAR SYSTEMS», n.º 3, págs. 856-869, 1986. DOI: 10.1137/0907058. arXiv: fld.1 [DOI: 10.1002].

[21] S. Fan, «An Introduction to Krylov Subspace Methods», págs. 1-14, 2018. arXiv: 1811.09025. dirección: http://arxiv.org/abs/1811.09025.

[22] Y. Vinay, S. N. Mojdeh y L. Ash (Chang), «Understanding the Bi Conjugate Gradient Stabilized Method (Bi-CGSTAB)», pág. 43, 2016.

[23] *The PYPL PopularitY of Programming Language Index is created by analyzing how often language tutorials are searched on Google*, https://pypl.github.io/PYPL.html, Accesado: 24-oct-2019.

[24] *Numba*, http://numba.pydata.org/, Accesado: 24-oct-2019.

[25] D. Das, M. Roy y T. Basak, «Studies on natural convection within enclosures of various (non-square) shapes – A review», *International Journal of Heat and Mass Transfer*, vol. 106, págs. 356-406, 2017, ISSN: 0017-9310. DOI: 10.1016/j.ijheatmasstransfer.2016.08.034. dirección: http://dx.doi.org/10.1016/j.ijheatmasstransfer.2016.08.034.